REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* 08-15-2007	2. REPORT TYPE final	3. DATES COVERED *(From - To)* 15 May, 2005 - 15 Aug, 2007

4. TITLE AND SUBTITLE Computational Methods for Feedback Controllers for Aerodynamics Flow Applications	5a. CONTRACT NUMBER FA9550-05-C-0048
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Siegel, Stefan G.; Seidel, Jürgen J.; McLaughlin, Thomas E.; Cohen, Kelly; Forsythe, James R.; Strang, William Z.	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) USAFA Dept of Aeronautics, HQ USAFA/DFAN, 2354 Fairchild Drive, Suite 6H22, USAF Academy, CO 80840-6222; Cobalt Solutions, LLC, 4636 New Carlisle Pike, Springfield, OH 45504	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) USAF/Air Force Office of Scientific Research, 875 N.Randolph St, Suite 325, Arlington, VA 22203-1768	10. SPONSOR/MONITOR'S ACRONYM(S) USAF/AFOSR
Scott Wells	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution Statement A. Approved for public release; distribution is unlimited.

AFRL-SR-AR-TR-07-0336

13. SUPPLEMENTARY NOTES
Report a result of a Small Business Technology Transfer program

14. ABSTRACT
This report covers the final results of the STTR: "Computational Methods for Feedback Controllers for Aerodynamics Flow Applications." It is intended to be a comprehensive summary of the project. The goal of the project was to provide a robust, easy to use computational tool for developing feedback controllers for aerodynamic flow applications – i.e. a tool to develop closed loop flow control methods. The two major partners of the STTR were the USAF Academy's Department of Aeronautics, and Cobalt Solutions, LLC (CSLLC).

15. SUBJECT TERMS
flow control CFD feedback POD

16. SECURITY CLASSIFICATION OF: a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
UU	UU	UU	UU		James R. Forsythe
					19b. TELEPHONE NUMBER *(Include area code)* 603-822-2588

20070917131

Computational Methods for Feedback Controllers for Aerodynamics Flow Applications

Contract: FA9550-05-C-0048
STTR Final Report
15 August, 2007

Executive Summary

This program succeeded in achieving the goal set in the original proposal solicitation, which is the development of a software toolbox to enable feedback flow controller development. Our toolbox is comprehensive in that it covers all aspects of controller development. Starting with data collection of the unforced and open loop forced flow response, through low dimensional model development, controller design and testing all the way to controller verification against a truth model, all important aspects of controller development are covered. Since all feedback control related software is implemented in Matlab, the controller developer can easily implement any control algorithm conceivable, using one of the most popular state of the art 4th generation programming languages. At the same time, computationally intensive algorithms were parallelized in order to improve execution speed and allow for processing of even the largest data sets. This was done using the highly efficient open source file format hdf5, in order to allow low level access to the data by the user if needed. Based on feedback from industry representatives at the AFOSR contractors meeting in Long Beach earlier this month, we expect the toolbox to become an indispensable tool for controller development not just for academia, but also for applied problems encountered by the industry. The toolbox developed around the Cobalt flow solver allows for physically correct modeling of any flow control mechanism available today, by using rigid body motion, blowing and suction boundary conditions and body forces. This can be done both open loop, closed loop and even combinations of multiple controllers and different actuation mechanisms acting on a flow field at once can be investigated. These capabilities set the toolbox apart from any other commercially available toolbox, and also surpass the capabilities of existing research codes.

Personnel Supported

U.S. Air Force Academy	Cobalt Solutions, LLC
Stefan G. Siegel Jürgen J. Seidel Kelly Cohen Thomas E. McLaughlin, Selin Aradag Several USAF Cadets	James R. Forsythe William Z. Strang

Journal Articles and other peer reviewed work. For conference proceedings please see the references section.

Cohen, K.; Siegel, S.; McLaughlin, T.; Gillies, E.; Myatt, J., 'Closed-loop approaches to control of a wake flow modeled by the Ginzburg-Landau equation', Computers and Fluids, Vol. 34 Issue 8, September 2005

Stefan G. Siegel, Kelly Cohen, Thomas McLaughlin, 'Numerical Simulations of a Feedback Controlled Circular Cylinder Wake', AIAA Journal, vol. 44 no. 6, June 2006

Cohen, K.; Siegel, S.; McLaughlin, T., 'A Heuristic Approach to Effective Sensor Placement for Modeling of a Cylinder Wake', Computers and Fluids, Vol. 35, 2006

Haiqian Yu; Miriam Leeser; Gilead Tadmor; Stefan Siegel, 'Real-Time Particle Image Velocimetry for Feedback Loops Using FPGA Implementation', Journal of Aerospace Computing, Information, and Communication , vol. 3 no. 2, 2006

Suzen, Y.B., Huang, P.G., Jacob, J.D., Ashpis, D.E., "Numerical Simulations of Plasma Based Flow Control Applications," AIAA 2005-4633, June 6-9, 2005.

Siegel, Stefan; Cohen, Kelly; Seidel, Jürgen; Luchtenburg, Mark; McLaughlin, Thomas, 'Low Dimensional Modeling of a Transient Cylinder Wake Using Double Proper Orthogonal Decomposition', accepted for publication in J. Fluid Mechanics, June 2007

Stefan Siegel, Kelly Cohen, Jürgen Seidel, Thomas McLaughlin, 'State estimation of transient flow fields using Double Proper Orthogonal Decomposition (DPOD)', Active Flow Control, NNFM 95, R. King (Editor), Springer, pp. 105-118, 2007

Seidel, Jürgen; Cohen, Kelly; Aradag, Selin; Siegel, Stefan; McLaughlin, Thomas, 'Reduced Order Modeling of a Turbulent Three Dimensional Cylinder Wake', to be submitted to Computers and Fluids, Summer 2007.

Kelly Cohen, Stefan Siegel, Jürgen Seidel, Selin Aradag, and Thomas McLaughlin. 'Reduced Order Model Based Controller Design for Feedback Flow Control', Abstract to be submitted to Prog. Aerospace Sci., Summer 2007

Kelly Cohen, Stefan Siegel, Jürgen Seidel, Selin Aradag, and Thomas McLaughlin, 'Sensor based estimation of POD Flow States using Artificial Neural Networks', to be submitted to Computers and Fluids, Summer 2007

Seidel, J., Siegel, S., Cohen, K., and McLaughlin, T., "Feedback Control of a Circular Cylinder Wake", 3rd International Symposium on Integrating CFD and Experiments in Aerodynamics, USAF Academy, CO, June 20-21, 2007.

Kelly Cohen, Stefan Siegel, Jürgen Seidel, Selin Aradag, and Thomas McLaughlin, 'Reduced Order Model Based Controller Design for Feedback-Controlled Cylinder Wake', Abstract accepted for the 46th Aerospace Sciences Meeting and Exhibit, Reno, NV, 2008

Jürgen Seidel, Stefan Siegel, Kelly Cohen, Selin Aradag, and Thomas McLaughlin, 'Data Analysis of an Axisymmetric Bluff Body Wake using Fourier Transform and POD', Abstract accepted for the 46th Aerospace Sciences Meeting and Exhibit, Reno, NV, 2008

Selin Aradag, Stefan Siegel, Jürgen Seidel, Kelly Cohen, and Thomas McLaughlin, 'Filtered POD based Estimation of 3D Turbulent Separated Flows', Abstract accepted for the 46th Aerospace Sciences Meeting and Exhibit, Reno, NV, 2008

Stefan Siegel, Jürgen Seidel, Kelly Cohen, Selin Aradag, and Thomas McLaughlin, 'Open Loop Transient Forcing of an Axisymmetric Bluff Body Wake', Abstract accepted for the 46th Aerospace Sciences Meeting and Exhibit, Reno, NV, 2008

Table of Contents

1 Introduction

This report covers the final results of the STTR: "Computational Methods for Feedback Controllers for Aerodynamics Flow Applications." It is intended to be a comprehensive summary of the project. The goal of the project was to provide a robust, easy to use computational tool for developing feedback controllers for aerodynamic flow applications – i.e. a tool to develop closed loop flow control methods. The two major partners of the STTR were the USAF Academy's Department of Aeronautics, and Cobalt Solutions, LLC (CSLLC).

1.1 Identification & Significance of the Problem or Opportunity

Feedback flow control is an emerging discipline that applies the methods of closed loop feedback control to problems in fluid dynamics. Traditionally, these two research fields have existed independently and without mutual involvement. Control scientists have applied their knowledge historically to problems with a relatively small dimensionality. Single-input single-output systems of linear behavior are very well understood and theoretically covered. More recently, feedback control has been applied to more complex problems in structural dynamics as well as fluid-structure interaction. These problems are of higher dimensionality and typically can be described by a small number of dominant modes. They also often involve multiple sensors and actuators. Feedback control of relatively complex structures is currently state of the art in control sciences, and very few control researchers have ventured into problems of higher complexity. Even simple problems in fluid dynamics, however, present themselves to the control scientist as a system of much higher dimensionality whose governing equations are highly nonlinear and, for most boundary conditions of technical interest, cannot be solved in closed form. This poses a formidable challenge for the application of feedback flow control to fluid dynamics problems of technical interest. On the other hand, since most fluid dynamics problems feature local or global instabilities, feedback flow control holds great promise in improving these flow fields with very moderate control inputs, while yielding large reductions in flow quantities of technical importance like drag and unsteadiness of loads imposed by the flow field onto structures exposed to the flow.

The biggest challenge to be overcome in order to be able to make progress in the field of feedback flow control is in the area of modeling the dynamics of the flow field. The current work focuses on low dimensional modeling. This technique has been applied with great success to control problems involving structural dynamics. The goal is to capture the relevant dynamics of the system, be it a mechanical structure or a flow field, by a finite number of spatial modes. The state of the system is then characterized by these modes, and an observer or state estimator is usually employed to derive the state of the system in terms of the mode amplitudes based on sensor readings. The controller then acts on the mode estimates as opposed to the sensor readings. The advantage of this approach is in the ability to actually model the important features of the system without having to solve the governing equations directly or numerically. Furthermore, the model can be derived from either experimental or simulation data, thus allowing development of a controller based on the dynamics of the actual system that is to be controlled. Of the three described attempts at feedback flow control, the use of low dimensional modeling has shown the greatest promise to date.

For this reason, this project aimed at developing and marketing computational tools necessary to develop low dimensional models. Currently, no such tools are commercially available, and researchers interested in feedback flow control have had to develop their own tools from scratch. Since neither controls nor fluids experts are typically expert programmers, they need to expend significant time and effort to even obtain tools of limited functionality, typically only usable for a very specific problem. The aim of this STTR collaboration between the US Air Force Academy and Cobalt Solutions, LLC was to join the controls and fluids expertise at the Academy with the programming and numerical mathematical solution skills at Cobalt Solutions. This collaboration has resulted in a toolbox that provides an integrated low order modeling environment covering the entire range of tasks from fluid dynamics simulation through low order modeling, mathematical model development and controller design to controller implementation and testing of the controller design by applying it to a truth model based on closed loop fluid dynamic simulation.

2 Simulation Tool

The development of the simulation tool was conducted by Cobalt Solutions, LLC. The Simulation tool consists of the Computational Fluid Dynamics (CFD) solver and its interface routines to Matlab.

2.1 Overview of Cobalt CFD Solver

The CFD solver used is *Cobalt*, a commercial unstructured finite-volume code developed for solution of the compressible Navier-Stokes equations. The basic algorithm is described in Strang *et al.* (1999), although substantial changes/improvements have been made since this paper. The numerical method is a cell-centered finite volume approach applicable to arbitrary cell topologies (e.g, hexahedrals, prisms, tetrahdra). The spatial operator uses a Riemann solver, least squares gradient calculations using QR factorization to provide second order accuracy in space, and TVD flux limiters to limit extremes at cell faces. A point implicit method using analytic first-order inviscid and viscous Jacobians is used for advancement of the discretized system. For time-accurate computations, a Newton sub-iteration scheme is employed, the method is second order accurate in time.

For parallel performance, *Cobalt* utilizes the domain decomposition library ParMETIS (Karypis *et al.* 1997) to provide optimal load balancing with a minimal surface interface between zones. Communication between processors is achieved using Message Passing Interface (MPI), with parallel efficiencies above 95% on as many as 4000 processors.

The vast majority of Air Force vehicles operate at high Reynolds numbers where the flow is turbulent. The main methods for calculating turbulent flows with a CFD solver are Direct Numerical Simulation (DNS), Large Eddy Simulation (LES), and Reynolds-averaged Navier Stokes (RANS). The RANS approach is the most economical since it is designed to solve for the mean flow, but is subject to many modeling approximations. Since it models rather than resolves the bulk if not all of the turbulent motions this is an inappropriate choice for most flow control applications. DNS, on the other hand, makes no modeling assumption but is the most expensive approach since all turbulent motions must be resolved by the grid. Since the smallest scales of turbulence (the Kolmogorov length scale) decrease with Reynolds number, this approach is limited to low Reynolds number flows. LES is less expensive than DNS since it models only the small subgrid scales of motion and resolves the rest of the turbulent motions. However, since the "large" scales in the boundary layer are on the order of the boundary layer thickness (which is quite thin for high Reynolds number flows), this method is cost prohibitive at high Reynolds numbers for wall bounded flows.

Detached-Eddy Simulation (DES) is a hybrid technique first proposed by Spalart *et al* (1997) for prediction of turbulent flows at high Reynolds numbers (see also Spalart 2000). The motivation for developing DES was to combine the most favorable aspects of RANS and LES, i.e., the acceptable predictions using RANS models of thin shear layers (e.g., attached boundary layers) and LES for resolution of time-dependent, three-dimensional large eddies. For natural applications of DES, RANS is applied in the boundary layer, while outside the boundary layer in the separated region, LES is used. An array of flows ranging from "building block" applications such as the flow over a cylinder, sphere, aircraft forebody, and missile base to complex geometries including full aircraft have been modeled (e.g., see Travin *et al.* 2001, Squires *et al.* 2001, Constantinescu *et al.* 2002, Forsythe *et al.* 2002, Hansen and Forsythe 2003, Constantinescu *et al.* 2003). These and other applications have been largely successful, illustrating DES capabilities in accurately resolving chaotic unsteady features in the separated regions along with a rational treatment of the attached boundary layers. Recent DES predictions of the flow around complex configurations (all using *Cobalt*) include the massively separated flow around an F-15E at 65° angle of attack reported by Forsythe *et al.* (2004) (the first eddy-resolving simulation of flow around a full aircraft configuration), transonic shock-separated flow over an F/A-18E by Forsythe and Woodson (2003), and vortex breakdown on an F-18C by Morton *et al.* (2003, 2004).

2.2 Feedback Control Interface

For the STTR phase I, the interface between the CFD solver (*Cobalt*) and the *Matlab* controller was handled via an HDF v4 file. Both programs would pass data into the flow control file, and change status flags to allow for handshaking. However, this method proved problematic due to network file buffering limitations. The method is unreliable when the *Matlab* controller and *Cobalt* are running on different machines, with the interface file on a network drive. Research was conducted into file-locking the interface file, but no reliable solution was found. After several alternatives were explored and discussed, it was decided to re-write the *Cobalt* control interface to directly call *Matlab* from within the solver. This makes the communication method purely network based, rather than file based which improves both reliability and speed. This required a re-write of the Cobalt control interface as well as a re-write of several of the *Matlab* modules. Testing was conducted to ensure the new interface gave the same answers as a previous controlled case. The method has been highly reliable (no hand-shaking failures), and has sped up some simulations by as much as a factor of two, since file I/O is no longer required. An additional advantage is that the handshaking between *Matlab* and *Cobalt* can be done on a finer grain level as *Matlab* variables can be directly accessed within *Cobalt* at any time.

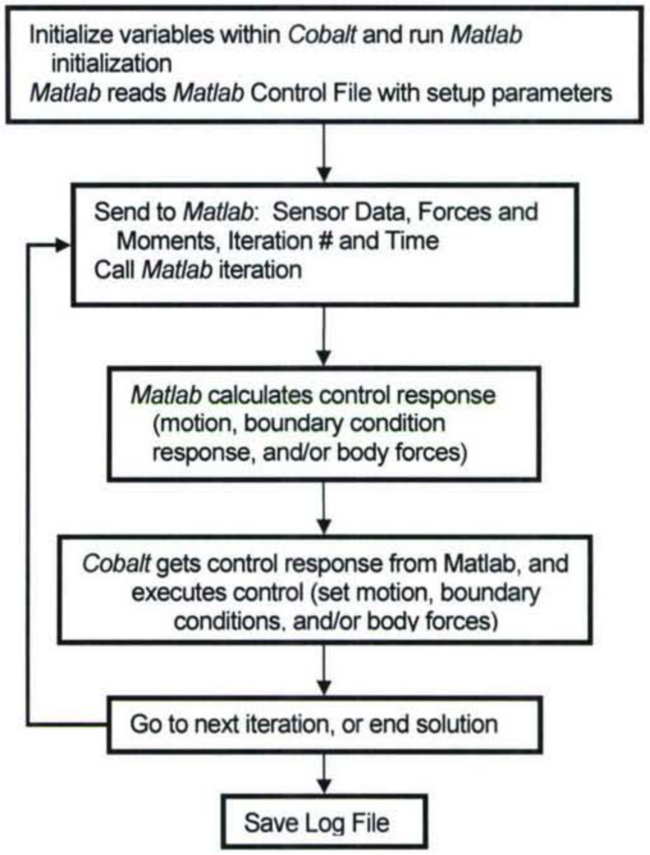
The toolbox allows for an unlimited number of sensors to be placed within the computational domain. At any sensing location, the following flow variables will be available to the controller:

- All three velocity components (U, V, W)
- Pressure
- Density
- Vorticity components

Additionally, the following global flow properties are available to the controller and can be used for closing the feedback loop:

- X,Y,Z components of force acting on the body
- X,Y,Z components of moment

A flow chart of the *Cobalt/Matlab* interface is shown in Figure 1. *Cobalt* controls the process, calling *Matlab* directly with its external interface routines, using the *Matlab* engine libraries. The underlying communication library that *Matlab* uses is pthreads. *Cobalt* takes care of communicating the *Matlab* data to all Cobalt processes using MPI (message passing interface).


Initialize variables within *Cobalt* and run *Matlab* initialization
Matlab reads *Matlab* Control File with setup parameters
Send to *Matlab*: Sensor Data, Forces and Moments, Iteration # and Time
Call *Matlab* iteration
Matlab calculates control response (motion, boundary condition response, and/or body forces)
Cobalt gets control response from Matlab, and executes control (set motion, boundary conditions, and/or body forces)
Go to next iteration, or end solution
Save Log File

=

Figure 1: Flow chart of *Cobalt/Matlab* interface

2.3 Actuation Capabilities

Cobalt currently contains three methods of actuation: rigid body motion, blowing and suction boundary conditions, and internal body forces. Each actuation type is described in the following sections.

2.3.1 Rigid Body Motion

A key feature in *Cobalt* is the capability of computing the flow around geometries undergoing rigid body motion. Simulation of rigid body motion is achieved through an Arbitrary Lagrangian Eulerian (ALE) formulation, where the grid is neither stationary nor follows the fluid motion. The conservation equations are solved in an inertial reference frame with modifications to the spatial operator in order that the advection terms are relative to the (non-inertial) grid reference frame. This requires simple modifications to many boundary conditions and to the initial conditions for the Riemann problem. The ALE formulation also requires modifications to the time-centered implicit temporal operator. A number of Newton sub-iterations are used to reduce errors associated with integrating over the time-step with an implicit temporal operator.

At each timestep, motion feedback is accomplished by the specification (within Matlab) of a 3x3 transformation matrix that specifies the orientation of the new grid location with respect to the lab reference frame. Additionally a three element vector is prescribed to handle translation. Thus the body is free to move with six degrees of freedom – three degrees in rotation, and three in translation.

A sample case using motion is seen in Section 4.3.

The version of Cobalt just released – V4.0 – adds overset capability so that multiple bodies can move relative to each other. In the future, the flow control tools will be modified so they can act independently on each grid desired.

2.3.2 Blowing and Suction Boundary Conditions

As part of this project, feedback boundary conditions were added. Control of multiple boundary condition patches simultaneously is possible. At each iteration, *Matlab* specifies for each patch either the massflow or the velocity. Additionally if massflow is specified, then total pressure and temperature must be also set. If velocity is specified, then static density, and the derivative of velocity with time are required. Technically the velocity specification can only be used for low subsonic cases due to assumptions made in its derivation.

A sample case using blowing/suction is seen in Section 4.2.

If open loop forcing is desired without using the Matlab interface, an additional boundary condition is available that provides a sinusoidally varying velocity or massflow. The user can set the frequency and the amplitude of the oscillation.

2.3.3 Body Forces

To enable plasma simulations, body forces within the CFD volume can be prescribed through the Matlab interface. Prior to coding the matlab interface, body forces were read routines were coded to read the prescribed body forces and interpolate them onto the unstructured grid. The method was validated by doing a code to code comparison using the prescribed body forces of Suzen et al. (2005). The body forces are shown in Figure 2. The resulting velocity is shown in Figure 3, and is a good match in peak magnitude and distribution to Suzen et al. (2005).

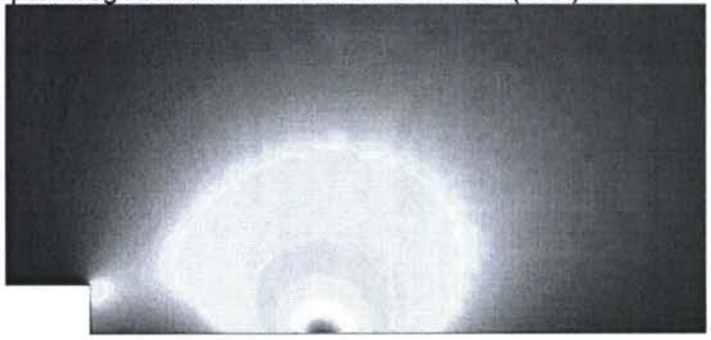


Figure 2: Prescribed body force magnitude in vicinity of the electrodes

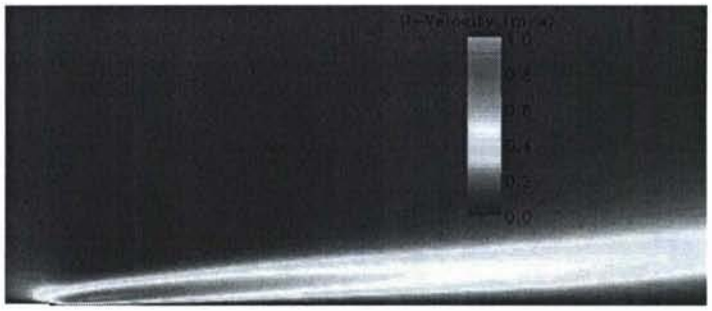


Figure 3: Computed U-velocity component behind electrodes.

For feedback control, Matlab specifies one or more controller. For each controller, a three-dimensional curvilinear structured grid is created, and a body force vector specified at each point. This data is passed to Cobalt, which then interpolates these values onto cell centers using inverse distance squared interpolation based on the Matlab structured grid cell that the Cobalt unstructured grid cell center falls into.

Careful attention was paid to designing a search algorithm to find where Cobalt cell centers fell within the Matlab grid. If there are N cells in Cobalt, and M points in the Matlab, then a brute force search would scale as NxM. The search method employed is similar to octree based search methods, but used a simple Cartesian grid. A Cartesian bounding box is drawn around each controller, then subdivided based on the number of points in the grid. Points of the control grid are then placed into bins based on their location within the Cartesian grid. The CFD cell centers are then located by determining the bin they fall in, and searching all control cells that bracket this bin based on point locations. This method proved to drop search times for large control grids from being larger than a CFD timestep, to a small fraction.

A sample of body forced open looped forcing is seen in Section 4.4.

2.4 Tap Data Output

The HDF interface for tap files was upgraded (re-written) to HDF version 5. This version allows files larger than 2GB, a crucial requirement to perform 3D calculations. Since HDF V5 is a new format, this required a complete re-write of these tap routines.

In order to facilitate the creation of POD modes for large 3-D cases, a tap extract capability was added to *Cobalt*. Normally to create POD's, either a set of flow taps are used in the simulation, or the entire flow solution is used from a flow visualization file. Using the tap procedure approach requires overhead at startup and during the run to locate the taps within the CFD grid, and to interpolate the data onto the taps. The interpolation also adds error to the results. The approach of using the entire flow viz file takes a significant amount of storage and the overhead of converting the flow viz files into a useful format (e.g. hdf). Tap extracts solves these problems by allowing the user to define regions in space (e.g. cylinder, cone, sphere), within which all data at the CFD points will be output to an HDF5 tap file. Since only a subset of the CFD solution needs to be output (e.g. the near wake behind the cylinder), the data is more compact than a full flow viz file. Additionally, no extra interpolation is required, and searches are very rapid.

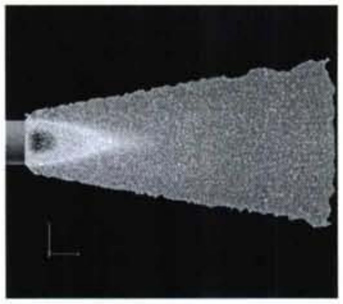


Figure 4: Cross plane of a "Cone" extract behind the axisymmetric cylinder.

One type of extract employed was an arbitrary cone, with an example shown in Figure 4. The method puts the tap variables at the points requested into the HDF5 tap file at the time intervals requested. Multiple extracts can be handled within the same HDF5 file. This file can then be used to create POD modes by using the *ppod* code, detailed below. A utility *hdf2fv* was also created to convert the tap extract HDF5 file into a viewable flow visualization file (Fieldview). *Hdf2fv* is also able to convert the resulting POD modes calculated by ppod into a flow visualization file. The types of extracts available are:

- Cone
- Cell (a large tetrahedral, prism, or hexahedra so that arbitrary geometries can be extracted)
- Surfaces (you may pick CFD boundary conditions to extract)

2.5 Parallelized POD Tools

In order to perform POD on large datasets, a parallelized POD module was required, to complement the POD tools developed within *Matlab*. This method was then extended to include the Double Proper Orthogonal Decomposition (DPOD) method described in Section 3.2.3. Finally code (hdf2fv) was developed to convert extracts, POD modes, or DPOD modes into a Fieldview file for visualization.

2.5.1 Parallelized POD Algorithm

A stand-alone Fortran Parallel POD code (PPOD) was written in order to enable rapid POD creation of large 3D datasets. The PPOD code is also required for large 3D datasets due to memory requirements – large datasets can easily break the 2Gb ram limit of a 32 bit machine. See Section 3.2.1 for a description of POD.

The code uses domain decomposition in space, splitting the tapped data nearly equally (within one tap) amongst the processors. File I/O is handled using parallel HDF in order to read the data as quickly as possible, and with each processor reading only the data required to keep the memory footprint low. The same algorithm as previously applied within *Matlab* is used to create the POD modes on each processor. On each processor the cross-correlation matrix is formed based on the taps on those processors. The contributions from all processors are then summed using the Message Passing Interface (MPI). On one ore more processors, the Scalable Linear Algebra PACKage (ScaLAPACK) is used to solve for the eigenvalues/eigenvectors of the cross-correlation matrix, with the matrix stored in packed form to reduce memory requirements (the matrix is symmetric). Results are then communicated back to each processor using MPI, and the POD modes are formed on each processor. The solution of the cross-correlation matrix can become time consuming for problems involving a large number of frames (the size of the matrix is NxN where N is the number of frames). Therefore the solution of the matrix is performed on multiple processors once it becomes large enough, using ScaLAPACK. Results are then written back to an HDF file using parallel HDF v5.

Benchmarks were performed using the PPOD code on a 3-D Cylinder at a Reynolds number of 3900 (Reference 4). A subset of the grid points was used as taps, with a total number of 226,543. 143 timesteps were used (one ever 5 iterations of the CFD run). The resulting performance is shown in Figure 5. The baseline memory used by the code on one processor was 133 Mb, while *Matlab* ran out of memory on a machine with 1Gb of Ram. There was therefore a substantial memory savings attributed to making a streamlined Fortran code. The figure then shows the memory at one processor divided by the memory at the given number of processors. This number indicates the factor by which the memory is reduced on each processor, which would ideally be equal to the number of processors (the ideal line). The memory reduction fails to follow the linear relationship at larger numbers of processors since there is a fixed amount of memory per processor to store the correlation matrix – proportional to the number of timesteps squared. However, at 16 processors a scant 13Mb is being used per processor, so much larger problems could be handled on cluster which typically contain 2Gb per processor.

Figure 5 also shows the speedup in CPU time versus number of processors, prior to the addition of ScaLAPACK for parallel computation of the cross-correlation matrix. The speedup is taken relative to the dual processor performance since these are dual processor nodes, and there is a certain amount of overhead involved in running on two cpus per node. The performance of the code peaks at 12 processors with a factor of almost 10 speedup. At this number of processors the total computation time, including file read, is 38 seconds, compared to the five minutes that *Matlab* took. At 12 processors 18% of the time is used in the communication of the cross-correlation matrix (which in this example was done on a single processor), while 8% of the time is used for file I/O. The addition of ScaLAPACK has further reduced the processing time for the cross-correlation matrix.

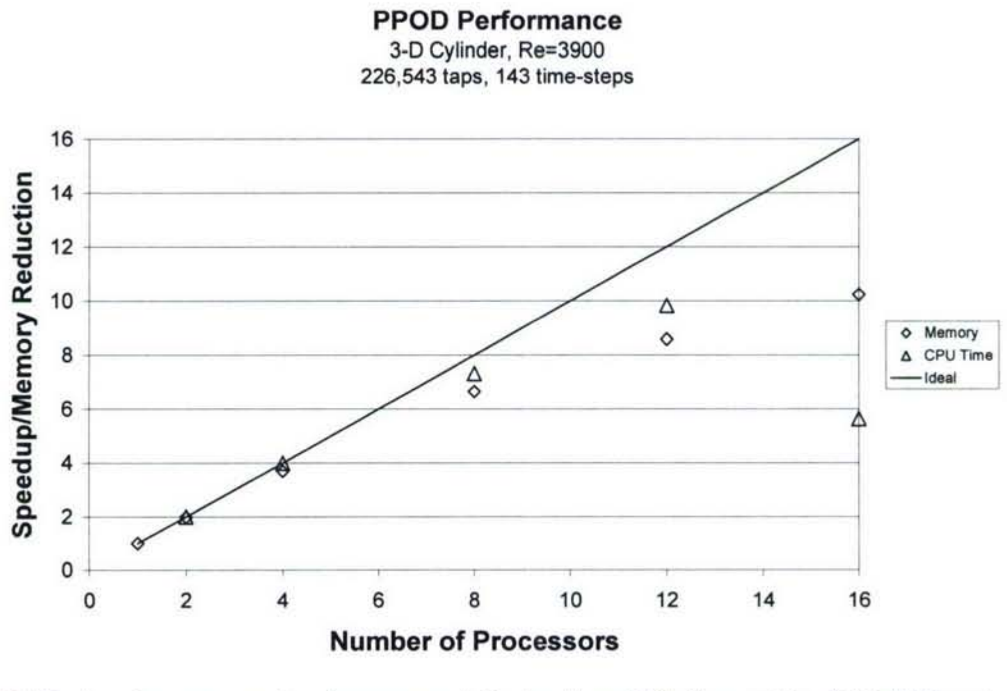


Figure 5: Parallel POD code performance vs. number of processors used. Benchmark on a 2.8Ghz Xeon machine with Gigabit Ethernet.

A sample of this method applied to the axisymmetric cylinder is shown in Figure 6, using the extract shown in Figure 4. The extract contained 100,000 points (compared to 1.5M cells for the full grid), and used 1000 time samples. Creation of the modes by *ppod* took two minutes on 12 processors. About ¼ of the time was used in reading the 1.4GB file, ¼ of the time spent in communication, and a bit less than half the time in inverting the C matrix (1000x1000) on a single processor (prior to the addition of ScaLAPACK). Working on this case, it was discovered that the read performance of HDF5 on the tap files was extremely slow in parallel. This was remedied by increasing the chunk size in time of the data, leading to increased performance, but an increase in file memory used (potentially 9 extra timesteps).

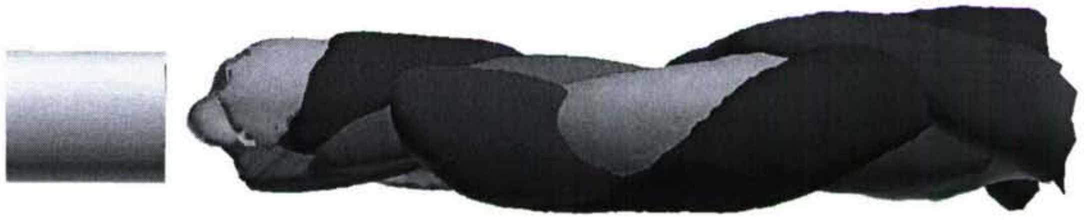


Figure 6: Isosurfaces of Mode 1-4 on the axisymmetric cylinder using tap extracts, ppod, and hdf2fv.

2.5.2 Parallelized DPOD Algorithm

The Double Proper Orthogonal Decomposition method described in Section 3.2.3 has been implemented in the PPOD code, as described. The user must first segment the input data based on iteration numbers so that a Short term POD (SPOD) can be performed (see Section 3.2.2). That is, bins must be described based on a beginning and end iteration number for each bin. These iteration numbers are then entered into the PPOD job file. The user then also specifies the number of DPOD modes to be calculated (main mode and shift modes). PPOD will then first calculate the SPOD modes, then calculate DPOD modes from these. Both sets of modes are stored in the resulting HDF file. The DPOD modes can be left un-normalized, or be normalized by type or by energy.

During processing the PPOD code makes an attempt to sort the SPOD modes so that each bin is in the same order, making the DPOD calculation meaningful. It does this by doing a spatial correlation between each mode in one bin, and the neighboring bin. It then sorts the modes to give the best correlation between the two bins. Bins with a strong negative correlation are "flipped" – i.e. multiplied by -1. This process does not always work, especially during transients were correlations may be very weak. To remedy this, the user can specify manual ordering for any of the bins. Typically a user would calculate the SPOD modes, then examine the energy for each bin to ensure they stay in order for all bins. When they get out of order, you can backtrack to the first bin where they swapped order, and force it back into the appropriate order. An example of this process is shown in Appendix A.

2.5.3 POD visualization (hdf2fv)

The final tool available is a code (hdf2fv) to convert the HDF files over to fieldview format for viewing. This is a simple command line fortran code that works on either the tap extract hdf files, or the PPOD output HDF file. It will detect if it is converting an extract file or POD file automatically. The resulting fieldview file will contain all the CFD points of the original grid, with a variable "Blank" to specify what points were not included in the extract. The resulting fieldview file can then be used with the fieldview grid file for viewing. For POD files, the code can be applied to convert either the SPOD modes (or POD in the case of one bin), or the DPOD modes. Each SPOD bin, or each DPOD shift mode is dumped out as a separate file. This allows rapid examination of the bins or modes in fieldview by transient animation.

3 Modeling, Estimation and Control Tools Development

Feedback flow control is an emerging discipline which aims at applying the methods of closed loop feedback control to problems in fluid dynamics. Traditionally, these two research fields have existed independently and without mutual involvement. Control scientists have applied their knowledge historically to problems with a relatively small dimensionality. Single input single output systems of linear behavior are very well understood and theoretically covered. More recently, feedback control has been applied to more complex problems in structural dynamics as well as fluid-structure interaction. These problems are of higher dimensionality and typically can be described by a small number of dominant modes. They also often involve multiple sensors and actuators. Feedback control of relatively complex structures is currently state of the art in control sciences, and very few control researchers have ventured into problems of higher complexity. Several applications of closed-loop control have been reported in literature. Specific areas of interest include flow-induced cavity resonance (Cattafesta, 2003), vectoring control of a turbulent jet, separation control of the NASA Langley hump model (a variation on the Glauert Glas II airfoil) control of the vortex motion in the combustion recirculation region and control of vortex shedding in circular cylinder wakes.

3.1 Modeling Approaches

A closed-loop flow control system is comprised of a controller that introduces a perturbation into the flow, via a set of actuators, to obtain desired performance. Furthermore, the controller acts upon information provided by a set of sensors. There are three basic approaches to closed-loop flow control of a two-dimensional wake as depicted in Figure 7.

Model Independent Approach - involves the introduction of sensors in the wake and using a control law (usually linear) which produces a command to the actuator that forces the flow. The advantage of this approach is to show:

- No model of the flow field is required for controller design.
- Direct feedback eliminates the need for a state estimator.
- A simple control law may be implemented in an experimental set up with relative ease.

For example, let us consider the circular cylinder wake problem. Experimental studies show that a linear proportional feedback control based on a single sensor feedback is able to delay the onset of the wake instability, rendering the wake stable at Re about 20% higher than the unforced case. Above Re = 60, a single-sensor feedback may suppress the original mode but destabilizes one of the other modes (Roussopoulos, 1993). This approach is relatively simple to implement experimentally. However, the results are very limited for the challenging problem of an absolutely unstable wake.

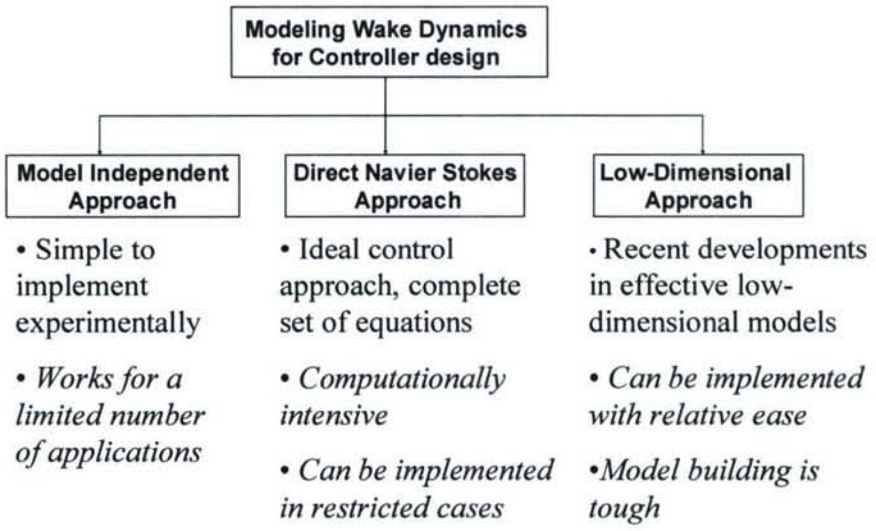

Modeling Wake Dynamics for Controller design
Model Independent Approach
• Simple to implement experimentally
• Works for a limited number of applications
Direct Navier Stokes Approach
• Ideal control approach, complete set of equations
• Computationally intensive
• Can be implemented in restricted cases
Low-Dimensional Approach
• Recent developments in effective low-dimensional models
• Can be implemented with relative ease
•Model building is tough

Figure 7: Approaches to Closed-Loop Flow Control

Direct Navier Stokes Approach – This approach is more structured as it applies conventional and proven model-based control strategies such as optimal control theory for flow control problems. Abergel and Temam (1990) developed conditions for optimality for a few simple applications. However, real time implementation of this approach to the cumbersome unsteady Navier-Stokes equations is not practical.

Low-Dimensional Approach - Low-dimensional modeling is a vital building block when it comes to realizing a structured model-based closed-loop strategy for flow control. For control purposes, a practical procedure is needed to break down the velocity field, governed by Navier Stokes partial differential equations, by separating space and time. A common method used to substantially reduce the order of the model is proper orthogonal decomposition (POD). This method is an optimal approach in that it will capture a larger amount of the flow energy in the fewest modes of any decomposition of the flow. The POD method may be used to identify the characteristic features, or modes, of a cylinder wake as demonstrated by Gillies (1998).

The major building blocks of this structured approach are comprised of a reduced-order POD model, a state estimator and a controller. The desired POD model contains an adequate number of modes to enable reasonable modeling of the temporal and spatial characteristics of the large scale coherent structures inherent in the flow though it may faithfully reproduce the flow. Further details of the POD method may be found in the book by Holmes, Lumley, and Berkooz (1996). A common approach referred to as the method of "snapshots" introduced by Sirovich (1987) is employed to generate the basis functions of the POD spatial modes from flow-field information obtained using either experiments or numerical simulations. This approach to controlling the global wake behavior behind a circular cylinder was effectively employed by Gillies (1998) and is also the approach followed in this research effort.

Low dimensional model development based on POD decomposition is a three step process, as shown in Figure 8. In the first step, data on the flow field to be modeled is gathered using either numerical or experimental methods. Selection of suitable data sets is a crucial step in model building and will be discussed in detail below. In the second step, spatial modes and their mode amplitudes are derived from the flow field data. The most often used approach is the "method of snapshots" developed by Sirovich (1987). Various methods of deciding how to precondition or cluster the data have been suggested in literature and are discussed below. In conjunction with the spatial POD modes, the associated mode amplitudes may be calculated using an inner product or least square fit approach.

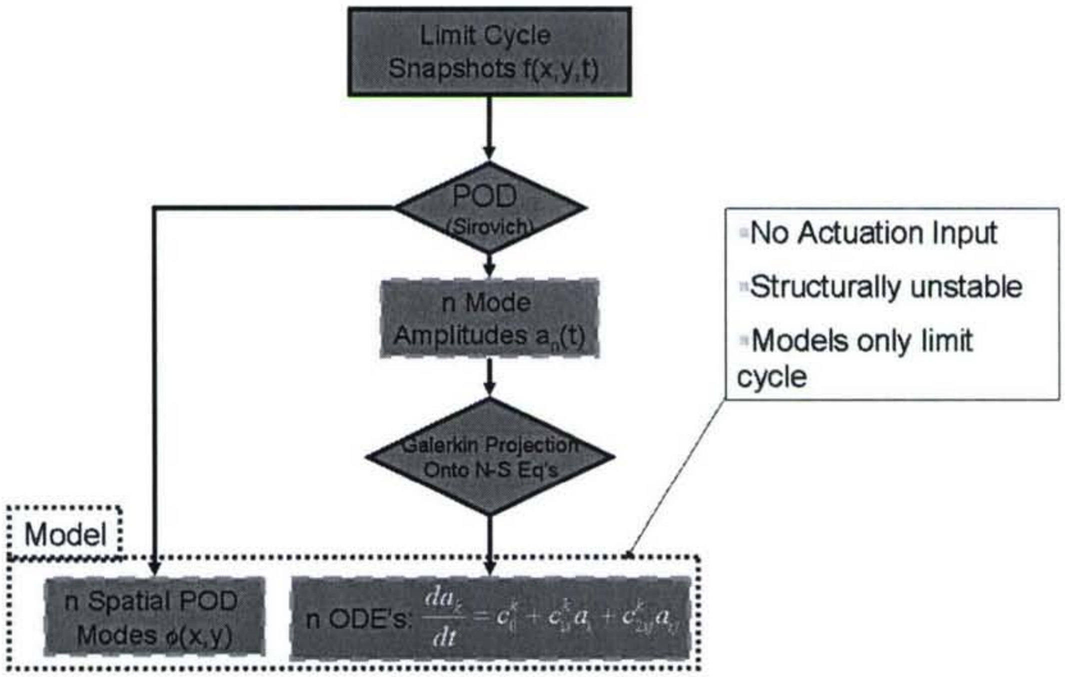

Limit Cycle Snapshots f(x,y,t)
POD (Sirovich)
n Mode Amplitudes $a_n(t)$
Galerkin Projection Onto N-S Eq's
No Actuation Input
Structurally unstable
Models only limit cycle
Model
n Spatial POD Modes $\phi(x,y)$
n ODE's: $\frac{da_k}{dt} = c_0^k + c_{1i}^k a_i + c_{2ij}^k a_{ij}$

Figure 8: Traditional low dimensional modeling approach.

To arrive at a low dimensional model, the mode set is truncated, usually based on an energy criterion (the eigenvalues in the Karhunen-Loève system represent twice the modal kinetic energy if POD is applied to the velocity field). The third and final step is the development of a model for the remaining mode amplitudes. This is commonly achieved by a Galerkin projection on the Navier-Stokes equations, which yields a system of equations that describes the evolution of the mode amplitudes over time. This set of equations can then be used to develop feedback control algorithms in a systematic fashion, or to test the performance of control algorithms against this model. All three steps of model development described above involve assumptions and potential problems, with many different solution proposed in literature, which we will discuss in the following.

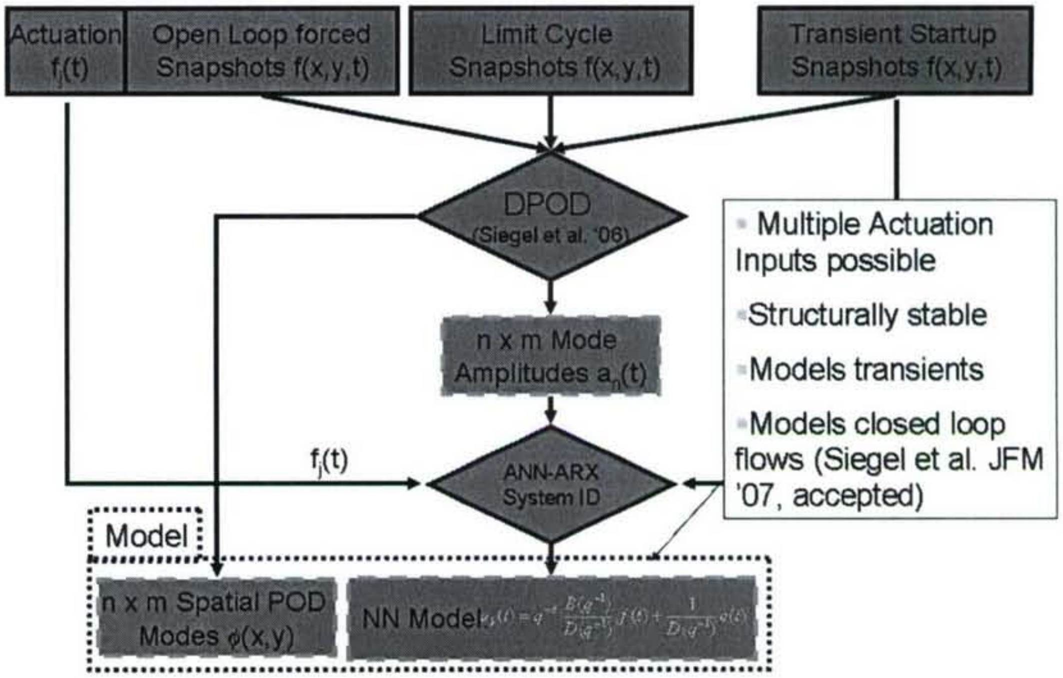

Actuation $f_j(t)$
Open Loop forced Snapshots f(x,y,t)
Limit Cycle Snapshots f(x,y,t)
Transient Startup Snapshots f(x,y,t)
DPOD (Siegel et al. '06)
n x m Mode Amplitudes $a_n(t)$
$f_j(t)$
ANN-ARX System ID
Multiple Actuation Inputs possible
Structurally stable
Models transients
Models closed loop flows (Siegel et al. JFM '07, accepted)
Model
n x m Spatial POD Modes $\phi(x,y)$
NN Model $a_k(t) = q^{-k}\frac{B(q^{-1})}{D(q^{-1})}f(t) + \frac{1}{D(q^{-1})}e(t)$

Figure 9: DPOD-ANN-ARX Modeling approach developed at the US Air Force Academy during this STTR program

Figure 9 gives an overview over the newly developed DPOD-ANN-ARX modeling approach developed within this STTR program. This overview flow diagram illustrates the modifications made to the traditional approach shown in Figure 8. At the input data level, transient data from both a change in Reynolds number as well as open loop forced flow states is included. Then, a modified POD procedure which we refer to as Double Proper Orthogonal Decomposition (DPOD) is employed. The main benefit of this method lies in the ability to derive spatial POD modes that cover a large range of flow conditions with small estimation errors, for details refer to the DPOD section in the following. Instead of the traditional Galerkin projection, a nonlinear ARX (Auto Regressive eXternal input) system identification method based on Artificial Neural Networks (ANN) is used in order to develop a dynamic model of the flow. We demonstrate for the circular cylinder wake that this approach yields a numerically stable model that is not just valid for the data used in its derivation, but also for a range of different Reynolds numbers, different open loop forcing conditions and, most importantly, feedback controlled flow states (Siegel et al. (2007)). Further details on the DPOD-ANN-ARX modeling approach are included in the following subsections of this report, while the modeling results are discussed in the following main section on Benchmark Flow Problems.

3.2 Spatial Mode Derivation

3.2.1 Basic Proper Orthogonal Decomposition (POD)

The Proper Orthogonal Decomposition (POD) of a two-dimensional scalar spatial field *u* evolving over time can be written as (Holmes, Lumley & Berkooz 1996)

$$u(x,y,t)=\sum_{j=1}^{\infty}a_j(t)\varphi_j(x,y)\,. \tag{1}$$

Here, *$a_j(t)$* are the mode amplitudes of the spatial modes *$\varphi_j(x,y)$*. For practical applications, formulating the problem using the Method of Snapshots (Sirovich 1987) proves to be advantageous. While this decomposition yields as many modes as there are snapshots in the original data set, it is typically possible to truncate the POD model at a relatively low number of modes while retaining most of the energy of the original flow field. This can be done by either inspecting the energy distribution in the modes, or by inspection of the spatial modes, which typically will not show any discernible structure beyond a certain mode number *j*. If POD is performed on the flow field without subtracting the mean flow, the first mode will be the mean flow, followed by modes representing the large scale fluctuations in the flow field. In the case of the cylinder wake, the largest fluctuating modes are the two modes representing the von Kármán vortex street. Since in almost all cases the modes obtained by POD describe the main features of the flow, we will refer to them as the *main modes*. The decomposition works particularly well for flow fields with large, time periodic features like the periodic vortex shedding in wake flows.

3.2.2 Short Time Proper Orthogonal Decomposition (SPOD)

Gillies (1995) and Siegel et al. (2005) have shown that for time periodic flows, modes identical to those obtained from snapshot ensembles containing a large number of shedding cycles can be obtained using snapshot ensembles of small integer number of cycles, down to a minimum of one shedding cycle. A similar idea to extend POD has been developed by Glezer et al. (1989) for flows that are not statistical stationary. Siegel et al. (2005) demonstrate that the difference between a spatial mode obtained from integer numbers of shedding cycles is minimal compared to a POD decomposition obtained from a large number (in the limit infinite) number of shedding cycles. Similar behaviour is observed in a fast Fourier transformation (FFT). While in an FFT the error due to non-integer numbers of cycles can be alleviated using windowing functions, this approach does not appear to work for POD decompositions (Siegel et al. 2005). Siegel et al. (2005) refer to POD of only a single oscillation cycle as Short time POD or SPOD, due to its similarity to procedures like Short Time Fourier decomposition.

SPOD allows for a decomposition of time evolving flow fields with some approximate periodicity into *(k)* individual events of exactly one cycle of the dominant frequency,

$$u^{(k)}(x,y,t)=\sum_{i=1}^{I}a_i^{(k)}(t)\,\varphi_i^{(k)}(x,y). \tag{2}$$

The result is a collection of *K* cycles in individual bins. Note that these bins may contain a different number of samples in time, or span slightly varying time intervals as the period of a cycle changes. However, since SPOD yields *K* bins of spatial POD modes that are valid for one individual cycle of a transient flow change, it is not as low dimensional as one would wish: The result of SPOD is one entire mode set for each period of the flow. It should also be noted that modes obtained from an individual cycle are *a priori* not orthogonal to modes from other cycles. In fact, if the data is completely periodic, the modes obtained from different bins are identical if the number of snapshots is constant per cycle.

3.2.3 Double Proper Orthogonal Decomposition (DPOD)

Building on the resulting spatial modes of an SPOD decomposition, one could conceive the following mode construction procedure: if the modes in two consecutive cycles vary only slightly, it should be possible to obtain a representation of the modes of the second cycle as the corresponding mode of the first cycle plus a small shift. This procedure, borne from the aforementioned "mean flow mode" or "shift mode" idea, can be formalized by realizing that mode *i* of all bins *(k)* from the SPOD procedure can be viewed as the input to a second POD as follows:

$$\varphi_i^{(k)}(x,y;t)=\sum_{j=1}^{J}\alpha_{ij}(t)\,\Phi_{ij}(x,y). \tag{3}$$

This leads again to an optimal representation of all SPOD main modes *i*. Equation 3.4 summarizes the double POD (DPOD) decomposition of the velocity field *u*:

$$u(x,y,t)=\sum_{i=1}^{I}\sum_{j=1}^{J}\alpha_{ij}(t)\Phi_{ij}(x,y). \tag{4}$$

This DPOD formulation takes the concept of the "shift mode" one step further: we can now develop a "shift mode", even a series of higher order shift modes, for all main modes *i* by applying the POD procedure to the POD mode sets $\varphi_i^{(k)}$. The resulting mode ensemble in its untruncated form has as many main modes I as there were snapshots in the smallest SPOD bin, and as many shift modes J as there were bins. It can then be truncated in both i and j, leading to a mode ensemble that is $I_M \times J_M$ in size. We will thus refer to the size of the truncated DPOD mode sets by indicating the truncation indices $I_M \times J_M$ in the following. A pictorial representation of the DPOD procedure is given in Figure 10. Starting in the top left corner, the data is split into K bins and each bin is used as an input data set for its individual POD procedure. The resulting SPOD modes are then collected *across the bins* and POD is applied again to obtain the shift modes.

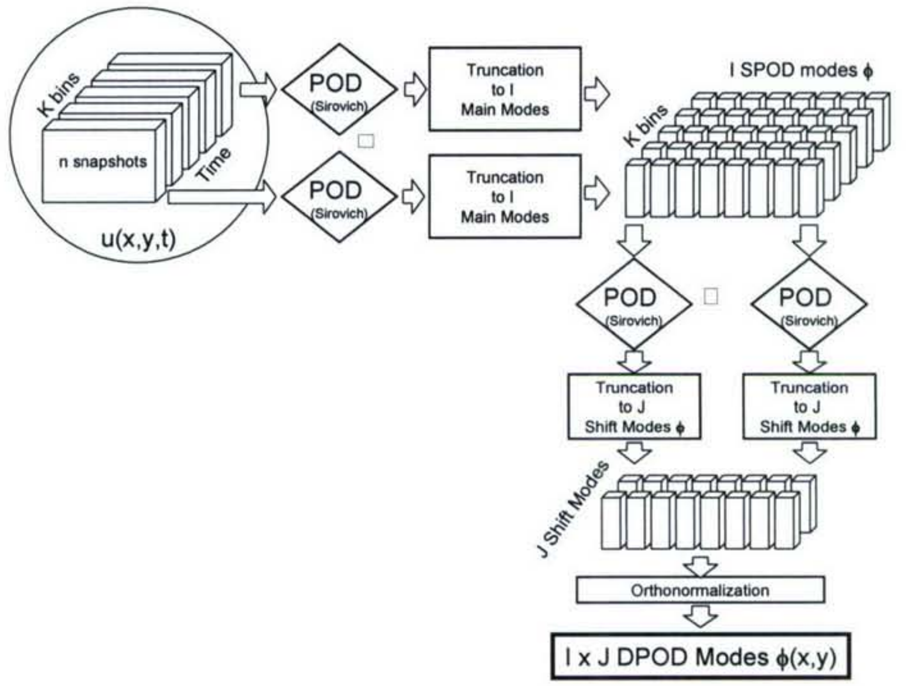


Figure 10: Flow chart of DPOD decomposition process.

The resulting Eigenfunctions can be truncated in both I and J in the same way as a regular POD decomposition. After orthonormalization, the decomposition is again optimal in the sense of POD. In the limit of J=1, the original POD decomposition is recovered. While the different modes distinguished by the index i remain the *main modes* described above, the index *j* identifies the transient changes of these main modes: For J >1, the energy optimality of the POD decomposition in that direction leads to modes that are the optimum decomposition of a given main mode as it evolves throughout a transient data set. If J = 2, then modes $\Phi_{1,1}$ and $\Phi_{1,2}$ are similar to the mean flow and its "shift mode" or "mean flow mode" as described by Noack et al. (2003) and Siegel et al. (2003), respectively. Thus the modes with indices j>1 can be referred to as first, second and higher order "shift" modes that allow the POD mode ensemble to adjust for changes in the spatial modes. We will refer to all of these additional modes obtained by the DPOD decomposition as *shift modes*, since they modify a given *main mode* to match a new flow state due to either a recirculation zone length or formation length change. This may be due to effects of forcing, a different Reynolds number, feedback or open loop control or similar events. Thus, in the truncated DPOD mode ensemble for each main mode, one or more shift modes may be retained based on inspection of energy content or spatial structure of the mode.

In the following section on Benchmark Flow Problems, we will demonstrate how this DPOD procedure can be used to create mode ensembles that cover both the unforced time periodic vortex shedding state of the circular cylinder wake, as well as the low amplitude forced flow within the lock-in region. This mode ensemble will thus cover not just the limit cycle, but also the influence of forcing onto the vortex shedding process.

3.2.4 Filtered POD for turbulent Flows

As the Reynolds number of the circular cylinder flow is increased beyond Re ~ 180, secondary instabilities lead to the formation of streamwise vortices (Williamson, 1996). While modeling these flow features will lead to additional DPOD modes, there is no apparent obstacle in applying POD based modeling to flow fields that contain both two and three-dimensional features, as has been show in the hybrid approach proposed by Ma & Karniadakis (2002). Extending their work to use the DPOD procedure introduced here, one could derive dynamic models capturing the effect of actuation and/or changes in Reynolds number for these types of flows. This could then be used to develop feedback controllers to suppress the von Karman type vortex shedding. Based on results of Cohen et al. (2003), where feedback of the von Karman mode only suppressed higher order harmonic modes, there is hope that suppressing the Karman vortex street might eliminate the streamwise vortices as well, since they are the result of a secondary instability that only exists in the presence of the von Karman vortex shedding. If this conjecture is in fact true remains to be shown.

At yet higher Reynolds numbers (Re > 3900), the von Karman vortices break down into smaller and smaller turbulent structures that ultimately dissipate their energy into heat. These smaller structures can be quite energetic and thus more and more POD modes will need to be retained in order to model a given fraction of the overall flow energy content. This behaviour of POD is due to the energy optimality of the procedure, and DPOD inherits this property from POD. As a result, both POD and DPOD models will become inherently large for flows that break down into turbulence. If the purpose of model development is feedback control, however, there may not be a need to model the small turbulent eddies in order to capture the dynamic behaviour of the large vortical structures. In the case of the circular cylinder wake, one may only be interested in modeling the von Karman type shedding for the reasons outlined in the previous paragraph. Thus, an approach where the flow data is subjected to either spatial or temporal filtering may be applied, as has been pursued by the authors with good preliminary results (Siegel et al. 2007). The approach proposed in this work removes small scale turbulent structures from the data used for POD mode derivation while retaining the large structures (i.e. von Karman vortices) that are of interest for feedback controller development. This approach is much like the use of spatial filtering in large eddy simulations employed in state of the art CFD solvers. With this approach, one introduces a choice of how much or how little of the smaller structures are included in the model. Thus, one can derive models with relatively few modes that nonetheless capture the dynamics of the flow that is relevant for feedback control. The DPOD-ANN-ARX approach is particularly suited to this type of modeling, since no turbulence model is required: As the entire model development is data driven and does not include projection onto the Navier Stokes equations, no closure equations are needed. The approach can thus be used as introduced here, with the only added step being a filtering process before the derivation of the DPOD modes. However, more detailed investigations into filter kernel type, size and cutoff wave length are needed.

3.3 Dynamic Model Development

3.3.1 Galerkin Projection

In the third step of model development, the resulting mode amplitudes from the POD procedure need to be described in terms of their dynamic behaviour using a set of equations. An established method to derive a set of equations based on POD modes is the use of a Galerkin projection, where the truncated POD results are projected onto the Navier Stokes equations. While mathematically valid, the resulting equations are a point design and the initial set of equations is often structurally unstable when integrated numerically (Noack et al. 2003). This structural instability has been described by Deane et al. (1991) as well as Rempfer (2000), who provides an explanation for this instability. Stuart (1958) describes a method to solve this instability problem, and recently developed models that take this approach into consideration to arrive at structurally stable models (Noack et al. 2003). However, for all Galerkin models the implementation of the actuation into the model is a major challenge, and is often limited to addition of a linear term that needs to be calibrated using experimental or computational data. The advantage of calibration is that it negates the need for spatial derivatives, which can cause problems when data uncertainty is present. Calibration has been successfully employed by Galletti, Bruneau, Zannetti & Iollo (2004) as well as Noack, Tadmor & Morzynski (2004b). For the scope of this work, we decided to use an alternative approach to Galerkin projection, which builds on decades worth of development performed by the controls and dynamic systems research communities, as described in the following sections.

3.3.2 System Identification – ARX Method

In an alternative approach to assuring model stability, the ARX (**A**uto **R**egressive e**X**ternal input) dynamic model structure, which is very widely used in the system identification community, is incorporated. A salient feature of the ARX predictor is that it is inherently stable even if the dynamic system to be modelled is unstable. This characteristic of ARX models often lends itself to successful modelling of unstable processes (Nelles 2001).

3.3.2.1 Linear ARX

System identification of a cylinder wake can be accomplished with a linear ARX model,

$$a(t) = q^{-d}\frac{B(q^{-1})}{D(q^{-1})}f(t) + \frac{1}{D(q^{-1})}e(t), \qquad (5)$$

where a(t) is the state vector representing the POD mode amplitudes $a_j(t)$ shown in Eqn.(1.1). f(t) describes the external input, which in the current effort is the vertical displacement of the cylinder and e(t) is the white noise vector. For the above case, B and D are matrix polynomials in q^{-1}.

The time delay operator is defined as

$$q^{-d}a(t) = a(t-d), \qquad (6)$$

where d is a multiple of the sampling period. The parameter matrix, θ, and the regression vector, φ(t), are respectively defined as

$$\theta = [d_{ij}\ b_{ij}]^T \qquad (7)$$

$$\varphi(t) = [a(t\text{-}1),\ldots,a(t\text{-}n),\ f(t\text{-}d),\ldots\ldots,f(t\text{-}d\text{-}m)]^T\ . \qquad (8)$$

As can be seen in Equation (5.4), the vector φ(t) is comprised of past states and past inputs.

The ARX predictor (Ljung 1999) may then be written as

$$\begin{aligned}\hat{a}(t \mid \theta) &= q^{-d} B(q^{-1}) f(t) + [1 - D(q^{-1})] a(t) \\ &= \varphi^T(t)\,\Theta.\end{aligned} \tag{9}$$

Eqn. (9) represents an algebraic relationship between the prediction, given on the left hand side, and past inputs and states, summarized by φ(t). The parameter matrix, θ, is determined during the estimation process. The main advantage of the ARX predictor is that it is always stable, even when the dynamic plant (the flow field in this case) being estimated is unstable. This feature is of utmost importance when modelling an unstable system such as the absolutely unstable cylinder wake flow.

It would be desirable to obtain a linear model that is able to meet all of the requirements described above. Unfortunately, limitations in the ability of linear systems to model a nonlinearly saturated limit cycle oscillation are to be expected. However, in some instances, certain non-linear systems may be successfully modelled using the addition of a set of virtual states (Ljung 1999). The authors spent considerable time in vain trying to apply state-of-the-art structured, numerically stable system identification techniques to this flow field. Linear state-space models having 25-90 states were developed and we concluded that the predictive capability is insufficient for feedback control as detailed by Cohen et al. (2006). Consequently, we decided to look into universal nonlinear approximators, such as artificial neural networks (ANN), for their inherent robustness and capability to approximate any nonlinear function to any arbitrary degree of accuracy (Cybenko 1989).

3.3.2.2 Artificial Neural Network ARX

The ANN employed in this effort, in conjunction with the ARX model, is the mechanism with which the dynamic model is developed using the DPOD mode amplitudes obtained from the CFD simulation data. Nonlinear optimization techniques, based on the back propagation method, are used to minimize the difference between the DPOD mode amplitudes and the ANN-ARX models estimation of the same while adjusting the weights of the model (Haykin 1999). While ANN have many advantages as stated, there are also drawbacks. These however can be avoided by proper design and analysis approaches as described in the following. The input-output set needs to be carefully constructed so as to avoid problems such as numerical ill-conditioning, over-fitting the data during the learning stage, and ensuring that the data is not over-represented in one region on account of another region (Nørgaard et al. 2000). A major problem using ANN concerns stability analysis. One of the best methods to address this is using a Stochastic Robustness Analysis technique based on Monte Carlo simulations. This approach, while more time consuming than stability analysis for linear models, is more accurate than using linearized stability methods since it does not make any simplifying assumptions. However, this added numerical effort is small compared to the numerical cost of the CFD simulations.

A general representation of nonlinear system identification, based on a hybrid ANN-ARX approach (Nørgaard et al, 2000), may be written as

$$a(t \mid \theta) = g[\varphi(t), \theta] + e(t), \tag{10}$$

where θ is the matrix containing the weights of the ANN that are estimated by a back propagation algorithm using a training data set (Nørgaard et al., 2000), and g is the nonlinear mapping realized by the feed-forward structure of the ANN.

The ANN-ARX predictor can then be expressed as

$$\hat{a}(t \mid \theta) = g[\varphi(t), \theta]\,. \tag{11}$$

The basic methodology to design an ARX-ANN plant for dynamic modelling of the DPOD mode amplitudes is presented in Figure 13. The ANN-ARX algorithms used in this effort are a modification of the toolbox developed by Nørgaard et al. (2000). The modification extends the toolbox for use in simulations, as opposed to one step prediction, of MIMO (multi-input, multi-output) systems. A schematic representation of the feed-forward ANN-ARX network topology is presented in Figure 14. After the DPOD mode amplitudes were obtained from the CFD data as described in the previous section, a single hidden layer ANN-ARX architecture is selected. The training set comprised input-output data obtained from CFD simulations. The model is validated for off-design cases and if the estimation error is unacceptable, then the ANN architecture is modified. This cycle was repeated until estimation errors were acceptable for all off-design cases.

The ANN-ARX predictor is **inherently stable** because, although the modelling approach is nonlinear, the algebraic relationship between the prediction and past states and inputs is preserved. This is extremely important when dealing with nonlinear systems represented by PDEs like the Navier Stokes equations, since the stability problems are more severe than in linear systems. The ANN-ARX approach is an ideal choice when the system to be modelled is deterministic and the signal to noise ratio (SNR) of the data is good (Nørgaard et al., 2000).

The choice of the specific artificial neural network (ANN) architecture was based on two main design criteria. The first concerns the number of hidden layers. This was selected as one, i.e. a single hidden layer, since it is the simplest form that allows for a universal approximator (Cybenko, 1989) and its effectiveness for system identification problems has been shown by Nørgaard et al. (2000). The second decision concerns the number of nodes. If the number of nodes in the hidden layer is small, the resulting error is unacceptable. As the number of nodes is increased, the error is reduced at the cost of computational complexity until a number of nodes is reached beyond which no further improvement in error is observed. In this effort a parameter study was conducted to find the optimal number of nodes for a laminar circular cylinder wake. This led to the use of 15 nodes in the hidden layer. The resulting ANN has the following features:

Input Layer: Two network input parameters, the normalized cylinder displacement and the Reynolds number. In addition to these readings, in order to obtain a strong representation of the dynamics of the system, the input layer includes 12 past outputs of 3 modes, 6 past inputs for each of the two inputs (Reynolds number and cylinder displacement) as described in the toolbox developed by Nørgaard et al. (2000). The number of time delays for the past outputs was about half a shedding cycle. On the other hand, the number of time delays for the past inputs was about one fourth of a shedding cycle. The selection of time delays for past inputs/outputs was based on a sensitivity study which investigated the trade-off between estimation accuracy and network complexity. Therefore the final configuration of the input layer chosen includes three past outputs, namely, the first three main DPOD modes i=1, j=1; i=2, j=1; i=3, j=1 for twelve time delays; and inputs, namely, Re and Control input for the six time delays, and one bias input.. The total number of inputs to the net is therefore (see Figure 14):

$$\mathbf{N_{in} = 12 \times 3 + 2 \times 6 + 1 = 49} \tag{12}$$

Hidden Layer: One hidden layer consisting of 15 neurons. The activation function in the hidden layer neurons is the *tanh* function. A single bias input has been added to the output from the hidden layer.

Output Layer: 15 outputs, namely, the 15 states representing the DPOD mode amplitudes of the 5 $\times$ 3 DPOD spatial mode basis. The output neurons have linear activation functions.

Weighing Matrices: The weighing matrix between the input layer and the hidden layer is of size (49x15), whereas the weighing matrix between the hidden layer and the output layer is of size (16x15). These weighing matrices are initialized randomly before training.

Training the ANN: Back propagation, based on the Levenberg-Marquardt algorithm, was used to train the ANN using the toolbox of Nørgaard et al. (2000). The training data (1849 time steps) is comprised of output from two CFD simulations. One is the transient development of the limit cycle during the ramping of the Reynolds number from Re = 40 to Re = 100. The second data set is an open loop forced data set (design case #1, shown in Figure 8), where transient forcing with $f=f_0$ and A/D=0.25 is applied for 15 shedding cycles. These two data sets were concatenated into one hybrid dataset consisting of data from both simulations. Along with the 5x3 DPOD mode amplitudes, the Reynolds number as well as the cylinder displacement were provided as inputs to the network. The training procedure converged after 300 iterations. The comparison between original mode amplitudes and mode amplitudes for the circular cylinder wake as estimated by the model is shown in the section on Benchmark Flow Problems

3.4 Sensor Based Mode Amplitude Estimation

For any state based feedback control system, it is necessary to estimate the amplitudes of the flow states – mode amplitudes – by means of flow measurements. Different approaches have been developed over the years to achieve this. The schemes can be characterized by the nature of the mapping function, which can be linear (Least Squares (LSQ), Linear Stochastic Estimation (LSE)), or nonlinear (Quadratic Stochastic estimation (QSE), Artificial Neural Network Estimation (ANNE)). Furthermore, the estimation may include a time history of the sensor signals (Dynamic Stochastic Estimation (DSE), ANNE) or simply predict the flow state at the next instant of time based on the current sensor signals (LSE, QSE, LSQ). While the linear and dynamic schemes outlined here have been described in open literature and used in the past by various research teams, the artificial neural network based flow state estimation (ANNE) was developed during this program at USAFA.

3.4.1 Least Squares

One of the simplest approaches uses a least squares type fit of the sensor data to the mode amplitudes, in essence minimizing the difference between the actual flow state and its modal based estimate. While simple to implement and numerically robust, this estimation approach requires typically many sensors to deliver a moderately accurate flow state estimate. It is also non-deterministic in terms of computation time due to the iterative estimation approach, which makes it inapplicable for real-time estimation in experiments. We use it mostly for comparison to more sophisticated schemes in the context of this investigation.

3.4.2 Linear Stochastic Estimation (LSE)

The time histories of the mode amplitudes of the POD model are determined by applying the least squares technique to the spatial modes and the flow sensor data. The intent of the proposed strategy is that the velocity measurements provided by the sensors are processed by the estimator to provide the estimates of the first two temporal modes. The estimation scheme, based on the linear stochastic estimation (LSE) procedure introduced by Adrian (1977), predicts the temporal amplitudes of the POD modes from a finite set of velocity measurements obtained from the CFD solution of the flow. For each sensor configuration, velocity measurements, equally spaced at an appropriate time interval, were used. The mode amplitudes, a_i were mapped onto the measured sensor signals, u_s, as follows:

$$a_n(t) = \sum_{s=1}^{m} C_s^n u_s(t) \tag{13}$$

where m is the number of sensors and $C^n{}_s$ represents the coefficients of the linear mapping. The effectiveness of a linear mapping between velocity measurements and POD states has been experimentally validated by Siegel et al. (2004) . LSE constitutes an improvement over the least squares approach described in the previous section, in that it is deterministic once the matrix Cs has been developed. However, it shares all the other properties with LSQ.

3.4.3 Dynamic Stochastic Estimation (DSE) and Quadratic Stochastic Estimation (QSE)

Recently, other non-linear techniques have been introduced such as the quadratic stochastic estimation (QSE) proposed by Murray and Ukeiley as well as by Ausseur et al. as well as introduction of time delays to the LSE, refereed to as DSE in this paper, as examined by Debiasi et al. These two approaches are improvements over the basic LSE approach, where either a time history is supplied to the mapping matrix of LSE, or a quadratic term (i.e. second matrix of coefficients) is added in order to improve the performance of LSE. The results section shows comparison of our newly developed ANNE approach (see following section) to these techniques.

3.4.4 Artificial Neural Network Estimation (ANNE)

In this effort, our estimation method of choice is based on a nonlinear system identification approach described by Nelles using Artificial Neural Networks (ANN) and ARX models. For the mapping of velocity measurements provided by the sensors onto the mode amplitudes of the DPOD modes, the modified NNARXM (Neural Network Autoregressive, eXternal input, Multi output) algorithm, originally developed by Nørgaard et al., is used as ANNE (Artificial Neural Network Estimation).

The decision was to look into universal approximators, such as artificial neural networks (ANN), for their inherent robustness and capability to approximate any non-linear function to any arbitrary degree of accuracy. The ANN, employed in this effort, in conjunction with the ARX model is the mechanism with which the dynamic model is developed using the POD mode amplitudes extracted from the CFD simulation. Non-linear optimization techniques, based on the back propagation method, are used to minimize the difference between the extracted POD mode amplitudes and the ANN while adjusting the weights of the model. In order to assure model stability, the ARX dynamic model structure is incorporated. This structure is widely used in the system identification community. A salient feature of the ARX predictor is that it is inherently stable even if the dynamic system to be modeled is unstable. This characteristic of ARX models often lends itself to successful modeling of unstable processes as described by Nelles.

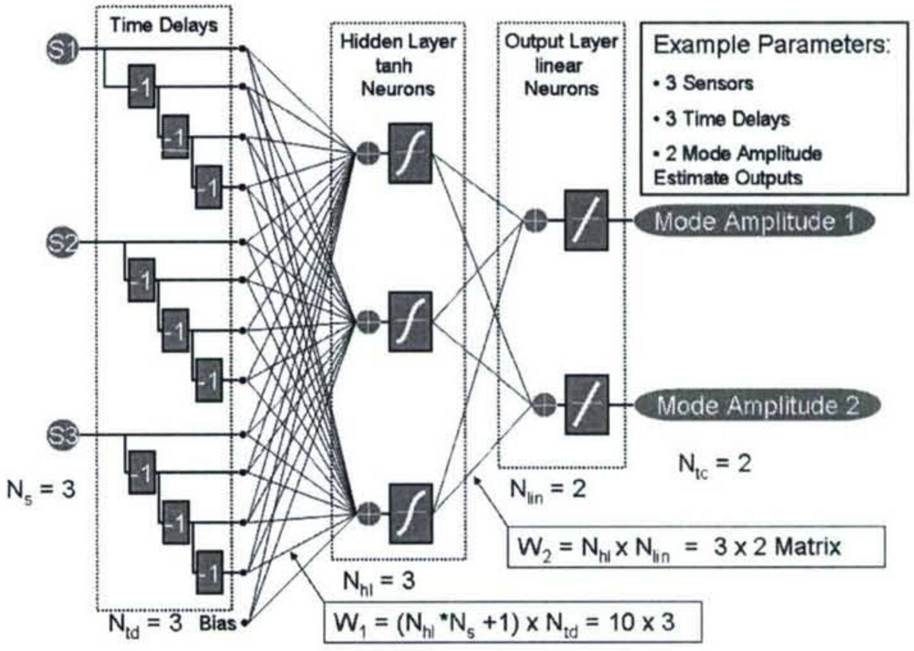

Time Delays
S1
S2
S3
-1
Hidden Layer tanh Neurons
Output Layer linear Neurons
Example Parameters:
• 3 Sensors
• 3 Time Delays
• 2 Mode Amplitude Estimate Outputs
Mode Amplitude 1
Mode Amplitude 2
$N_s = 3$
$N_{td} = 3$ Bias
$N_{hl} = 3$
$N_{lin} = 2$
$N_{tc} = 2$
$W_2 = N_{hl} \times N_{lin} = 3 \times 2$ Matrix
$W_1 = (N_{hl} * N_s + 1) \times N_{td} = 10 \times 3$

Figure 11: ANN network structure used for the ANN-ARX based Artificial Neural Network Estimator (ANNE)

Figure 11 shows a sample network topology that can be used for sensor based mode amplitude estimation. At the input level, three sensors are employed to provide data to the network. For each of these sensors, three time delays are used to provide a dynamic sampling in time of the sensor readings. The hidden layer with its neurons using a tanh activation function provides the nonlinear behavior of the estimator, while the output layer activation functions are linear. While in this example three sensors, three time delays as well as three hidden layer neurons are employed to estimate two mode amplitudes, all of these parameters can be adjusted independently in order to provide the necessary mapping from sensors to mode amplitudes. Defining these parameters involves an iterative approach where the complexity of the network is gradually increased until a desired estimation error is achieved.

ANNE, based on the Multilayer Perceptron Neural Network, uses an adequate training set as described in the results section. This data set depends on the estimation problem at hand and can be of different size. The ANN network is designed and then the "frozen" ANN design, with its associated weighing matrices, is validated with data not used for training. The purpose is to obtain a robust and real-time estimator for as low a number of sensors as possible for application to wake control. Once the network weights are fixed, the estimation can be performed in a time deterministic fashion, typically in real time for experimental applications.

The artificial neural network (ANN) developed for a laminar circular cylinder wake has the following features:

- **Input Layer:** No past outputs; 4 sensors each having: 4 past inputs each with "current time" signal + 3 time delays, i.e. ~ 1/8 of a shedding cycle per input. 65 Neurons (4 x 4 x 4 +1 bias = 65) in the input layer.
- **Hidden Layer:** One hidden layer consisting of 6 neurons. The activation function in the hidden layer is based on the non-linear *tanh* function. A single bias input has been added to the output from the hidden layer.
- **Output Layer**: Six outputs, namely, the 6 DPOD mode amplitudes: the mean flow, $\alpha_{1,1}$, the two fundamental von Kármán POD periodic modes, $\alpha_{2,1}$ and $\alpha_{3,1}$ and the three associated shift modes, namely, $\alpha_{2,1}$, $\alpha_{2,2}$ and $\alpha_{3,2}$. The output layer has a linear activation function.
- **Weighting Matrices**: The weighting matrices between the input layer and the hidden layer (W_1) and between the hidden layer and the output layer (W_2) depend on the number of sensors. For example, for the single sensor case W_1 is of the order of [65×6] and W_2 is of the order of [7×6]. These weighting matrices are initialized randomly.
- **Training the ANN:** Back propagation, based on the Levenberg-Marquardt algorithm, was used to train the ANN using the toolbox by Nørgaard et al. The training procedure converged in approximately 100 iterations. The training data is shown in the results section and contained 1500 snapshots. Sinusoidal forcing, at the fundamental Strouhal frequency and a constant amplitude of $y_{cyl}/D = 0.25$, is introduced only after the first 180 snapshots and stopped after exactly 15 full forcing cycles. The remaining 429 snapshots are then unforced.
- **Validating the ANN:** Six different validation data sets were used to represent the off-design cases as described in the results section. Each of the validation sets contained 1500 snapshots and includes sinusoidal forcing, at various frequencies and amplitudes

As shown in the results section of this report, excellent estimation quality can be achieved with relatively small networks. Due being a universal approximator, the ANNE method achieves typically better estimation results than any of the other approaches described in open literature, while requiring less sensors. This improvement of performance appears to be more pronounced the more nonlinear the mode amplitude time history becomes, and thus makes ANNE an ideal estimator for higher Reynolds number, turbulent flows where traditional estimation techniques become very inefficient.

3.5 Controller Development

While the main focus of this research was in the area of low dimensional model development, we have conducted various feedback controlled simulations, both using a single mode for feedback as well as using multiple modes for feedback. Both control approaches are defined in the following sections, while the results are presented in the Flow Benchmark Problems section.

3.5.1 Single Input Single Output (SISO)

The controller development is based on a Proportional and Differential (PD) controller. The full state estimator provides estimates for all 15 DPOD amplitudes which may then be used as input to a full state controller. Since a neural network model of the flow has been developed, several indirect control designs can be employed. The indirect design is very flexible and applicable in real-time for the problem at hand. Relevant concepts include approximate pole placement, minimum variance and predictive control. The approximation is based on instantaneous linearization, which is a popular method for control of ANN models.

Following the effort by Siegel, Cohen and McLaughlin, a similar PD feedback control strategy is employed for the single mode feedback control law:

$$y_{cyl} = K_{p21} \cdot \alpha_{21} + K_{d21} \cdot \frac{d\alpha_{21}}{dt} \tag{14}$$

Instead of directly specifying the $K_p$and K_d gains, these can be expressed in terms of an overall gain K and a phase advance φ for mode i:

$$\begin{aligned} K_{pi} &= K_i \cdot \cos(\varphi_i) \\ K_{di} &= \frac{K_i \cdot \sin(\varphi_i)}{2\pi \cdot f} \end{aligned} \tag{15}$$

with f being the natural vortex shedding frequency. Equation (14) refers to the single-input closed-loop control based on feedback using an estimate of Mode (2,1). We will refer to this control law as a Single Input Single Output (SISO) controller in the following, since only one mode is used for determining the feedback response.

3.5.2 Multi Input Single Output (MISO)

The control law above may be modified to enable dual-input from two modes, and a simple example is adding proportional control of the shift mode of the von Karman shedding mode, Mode (2,2). This control law which essentially constitutes a multi input single output (MISO) PD controller is:

$$y_{cyl} = K_{p21} \cdot \alpha_{21} + K_{d21} \cdot \frac{d\alpha_{21}}{dt} + K_{p22} \cdot \alpha_{22} + K_{d22} \cdot \frac{d\alpha_{22}}{dt} \tag{16}$$

where it is possible to substitute the proportional and differential gains in the same fashion as for the SISO controller, with an overall amplitude gain K and a phase shift φ for each of the two modes independently.

4 Benchmark Flow Problems

Several benchmark cases were chosen to exercise different portions of the toolbox. The laminar 2D cylinder received a lot of focus on model development, and uses motion as the control mechanism. The axi-symmetric bluff body uses blowing/suction as the control mechanism at the base and contains a fixed separation point. Additionally the added complexity of a three-dimensional flow and control is examined. The turbulent wake of a cylinder exercises portions of the toolbox for a more complex three dimensional problem. Finally, body forcing is applied to the circular cylinder as a means to examine the potential effectiveness of plasma actuators.

4.1 Laminar 2D Cylinder

Even at modest Reynolds numbers, the flow around these objects of relatively simple geometry can be of astounding complexity. A prime example of this behavior is the circular cylinder wake at laminar Reynolds numbers on the order of Re=100. While the famous von Kármán vortex street is easily observed and visualized (von Kármán, 1911), understanding and modeling its dynamics, with the eventual goal of controlling it, are daunting tasks.

Initially, the motivation for the development of these models was to gain physical understanding. Recently, the focus of reduced order model development has shifted from the desire to obtain a physical understanding of the flow field to use of the model for feedback flow control. In this context, a low-dimensional model that is computationally efficient is crucial for several reasons. For the development of control algorithms, a model is needed to aid in their derivation. State-of-the-art control theory can only derive controllers for problems with relatively few degrees of freedom. Once a control algorithm has been developed, for the implementation of the control algorithm in an experiment or application, it is necessary to solve the control equations in real-time. This can currently only be achieved for relatively low complexity controllers.

To achieve the necessary reduction in the order of the modelling problem, Proper Orthogonal Decomposition (POD), also referred to as the Karhunen-Loève decomposition, has been successfully used as a computational tool for reducing the order of the fluid dynamic system. Analyzing POD in detail, researchers recognized that it is not well suited to describe transient data sets, i.e. data sets that are neither stationary nor periodic in time, in its original form. Our newly developed Double POD (DPOD) solves this problem.

For the numerical simulations, the Cobalt was used. Boundary conditions at the far field are implemented as Riemann invariants. The cylinder surface is modeled as an adiabatic non-slip surface. Transverse body translation was used as forcing input to obtain open loop forced data sets. Rigid-body motion is achieved through an Arbitrary Lagrangian Eulerian (ALE) formulation, where the grid is neither stationary nor follows the fluid motion. The conservation equations are solved in an inertial reference frame, but the spatial operator is modified so that the advection terms are relative to the (non-inertial) grid reference frame. For all simulations presented here, a structured two-dimensional grid with 63700 nodes and 31752 elements was used. The grid extended from -16.9 cylinder diameters to 21.1 cylinder diameters in the (streamwise) x-direction, and ±19.4 cylinder diameters in (flow normal) y-direction, with the origin located at the cylinder centre.

Other pertinent simulation parameters are as follows:

- Reynolds Number: Re=100
- Damping Coefficients: Advection=0.01; Diffusion=0.00
- Non-dimensional time step: $\Delta t^* = \Delta t \cdot U/D = 0.05$

For validation of the CFD simulations of the unforced cylinder wake at Re=100, the value of the mean drag coefficient, c_d, was compared to experimental and computational investigations reported in the literature. At Re=100, experimental data, reported by Oertel (1990) and Panton (1996), point to c_d values ranging from 1.26 to 1.4. Simulations by Min & Choi (1999) obtained drag coefficients between 1.34 and 1.35. Using COBALT resulted in a value of c_{d0}=1.35, which compares well with the values in literature. Another important benchmark parameter is the Strouhal number ($St=f \cdot D/U$), which is the non-dimensional vortex shedding frequency for the unforced cylinder wake. Experimental results at Re=100, presented by Williamson (1996), show Strouhal numbers ranging from 0.163 to 0.166. The Strouhal number obtained in this effort is St = 0.163, which also compares well. The non dimensional time step in connection with the natural shedding frequency yielded about 100 CFD time steps per unforced shedding cycle. This was further reduced by down sampling to 20 snapshots per cycle which were then used as input to the DPOD procedure.

4.1.1 Application of DPOD to Cylinder Wake Data

4.1.2 Start-up Transient

The von Kármán vortex street develops as the result of global flow instability (Williamson 1996). This transient development of the oscillatory limit cycle starts when a closed recirculation zone forms downstream of the cylinder that is steady in time for Reynolds numbers smaller than Re_c=47. Increasing the Reynolds number, this unstable flow field starts to exhibit vortex shedding, which increases in strength over several shedding cycles as evidenced by the fluctuations in the lift force as shown in Figure 12. During this transient start-up, both the shedding frequency and the strength of the shed vortices change. By starting a CFD simulation at a Reynolds number below this threshold (Re=40 in our case), and subsequently ramping up the flow speed to the target Reynolds number of Re=100, one can study the details of the development of the limit cycle. The ramping was done smoothly using a constant acceleration, and no numerical artefacts during the transition were observed.

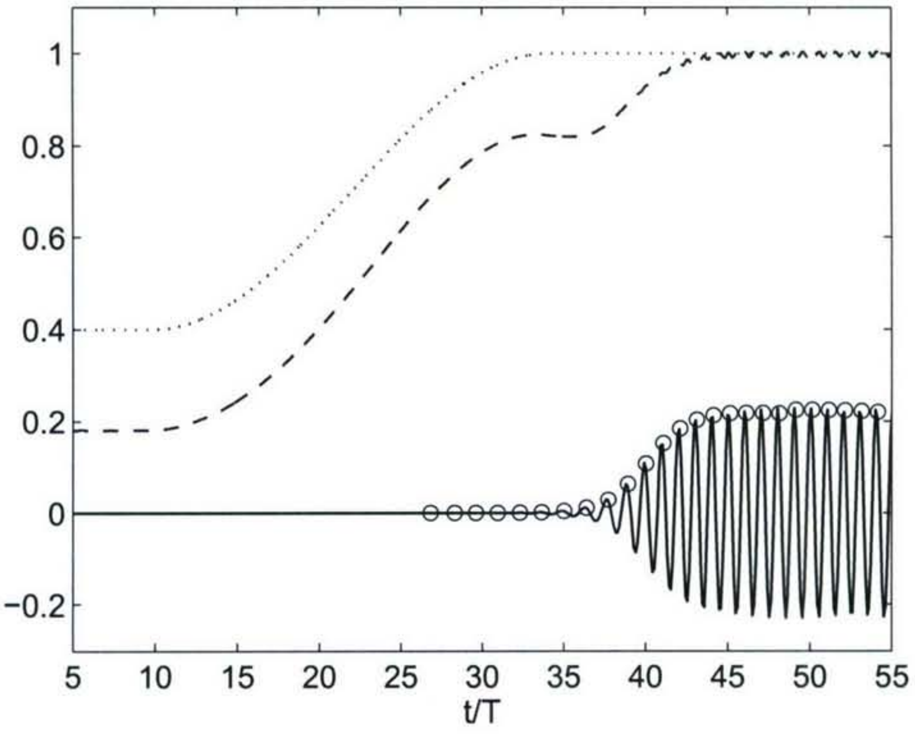


Figure 12: Normalized lift (C_L / C_{D0}, -) and drag (C_D/C_{D0}, --) coefficients during evolution of the limit cycle oscillation. The Reynolds number (Re / 100, ..) is ramped from 40 to 100 as shown. The SPOD segmentation times are indicated by circles.

The traditional approach to development of a set of POD modes uses data from one or several shedding cycles of the fully developed, nonlinearly saturated limit cycle, i.e. data from later than about t/T = 55 into the simulation shown in Figure 12. While this set of POD modes models the limit cycle extremely well, it can be seen in Figure 13 that an increasing estimation error results when data sets taken during the transient start-up of the shedding are mapped onto these spatial modes. The POD spatial mode ensemble was truncated here at 10 modes, but the estimation error remains unaffected when more spatial modes are retained. The reason for this large estimation error lies in the drastic changes of both the mean flow as well as the formation length during the transient start-up. The length of the recirculation zone, which extends initially to about 5 cylinder diameters downstream of the cylinder centre, shortens to about 2.5 cylinder diameters once the limit cycle is fully developed. The formation length (Williamson 1996), equivalent to the location of the maximum velocity fluctuations, changes in a similar fashion.

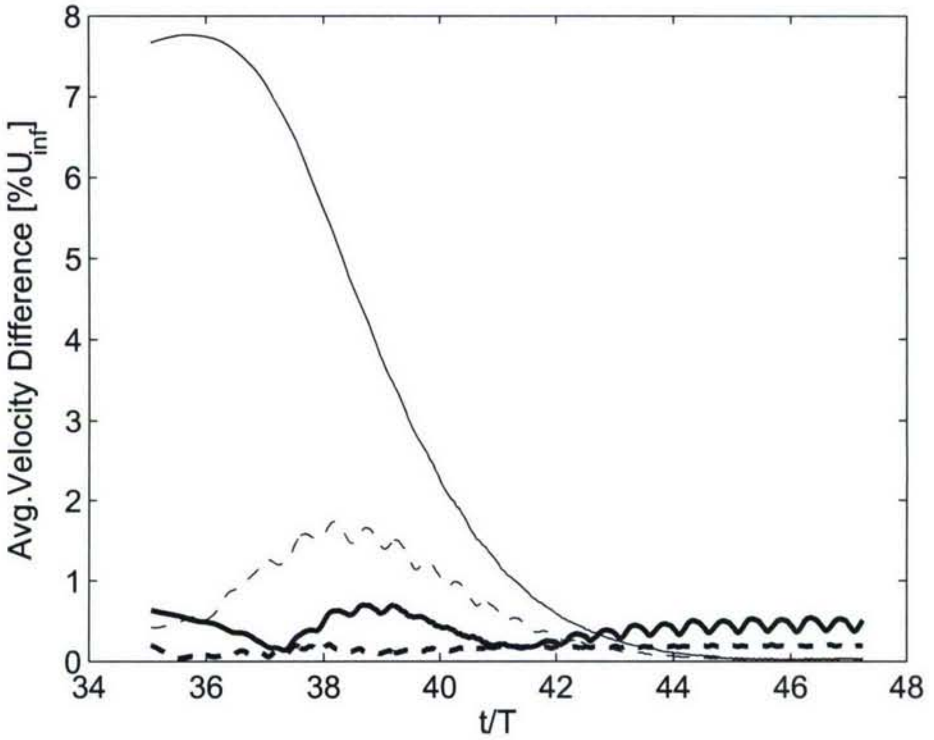
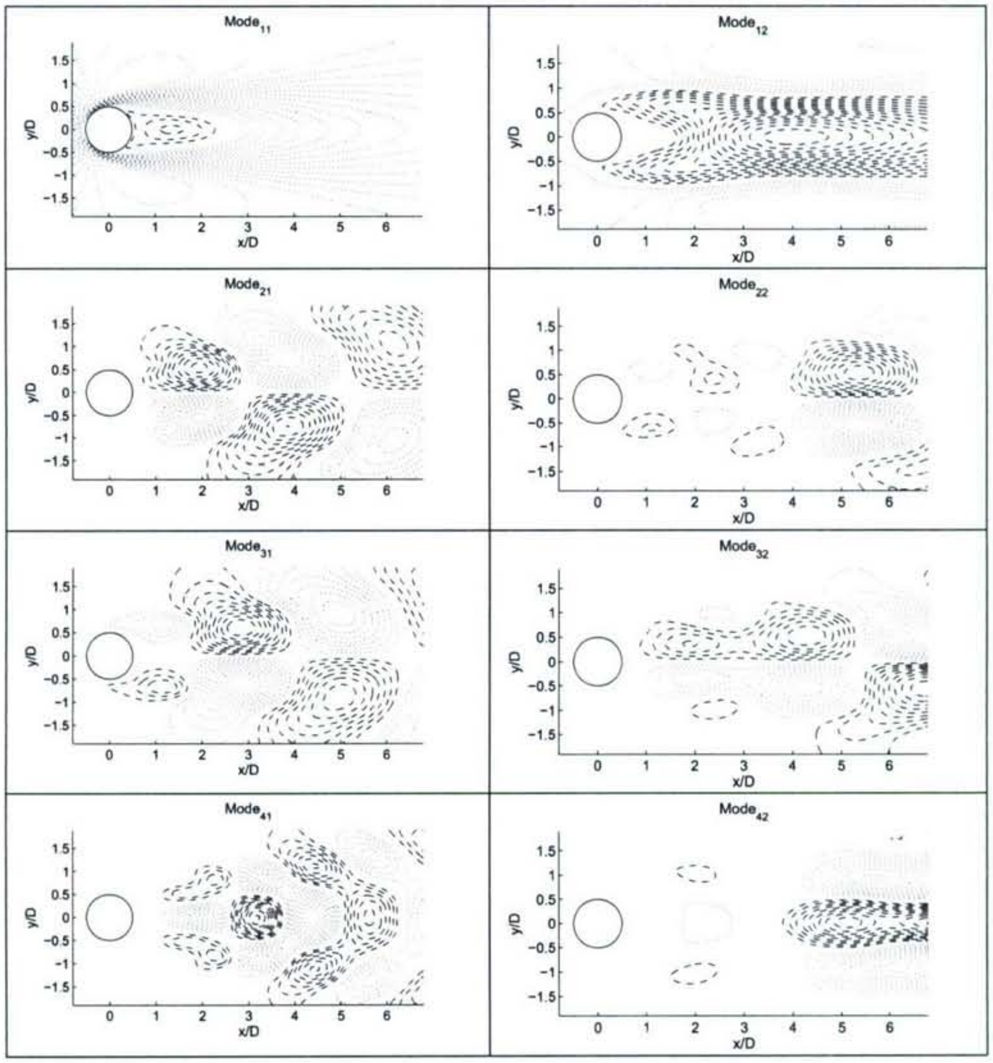


Figure 13: Average reconstructed flow field error of differently truncated mode sets for the simulation data of the transient development of the limit cycle. 10 POD modes derived from time periodic flow (thin solid line), 10 POD modes derived from time periodic flow plus shift mode (thin dashed line), 5×2 DPOD modes (thick solid line), 5×3 DPOD modes (thick dashed line).

The change in mean flow can be compensated for by the addition of an artificially created "shift" (Gerhard et al. 2003) or "mean flow" (Siegel et al. 2003) mode, which allows the length of the recirculation zone to adjust. This addition to the mode set greatly reduces the estimation error, as seen in Figure 13. However, the fluctuating modes are unchanged, and thus the formation length is only correctly modelled for the limit cycle. Since there is no unsteadiness initially, the model including the shift mode shows small errors both for the steady recirculation zone flow at the beginning of the start-up transient, and for the limit cycle that was used to derive the spatial POD fluctuating modes. In between these states, however, the model shows a significant error caused by the inability of the fluctuating modes to correctly track the change of the formation length during start-up. Since this exchange of energy between the fluctuating modes and the mean flow dominates the start-up dynamics, the POD model with shift mode only (referred to in Noack et al. (2003) as Model B) cannot possibly capture the dynamics of the start-up transient correctly, as was shown in Figure 12 of Noack et al. (2003). However, the effects of formation length change may be correctly included in a low dimensional model by adding modes that capture this behaviour. Noack et al. (2003) achieve this using stability modes derived from a global stability analysis of the steady solution of the flow. In this work, we will demonstrate a systematic approach to achieve the same effect, using the DPOD decomposition detailed below.

Processing the same data set using the DPOD procedure, one can obtain a 10 or 15 mode model by retaining five main modes and either one or two shift modes for each main mode. The spatial POD modes of the 5x2 mode model are shown in Figure 14. To obtain these modes, separate POD decompositions were performed for 11 independent snapshot ensembles consisting of exactly one shedding cycle each. The shedding cycles were determined using a peak detection algorithm applied to the lift force shown in Figure 12 and are indicated by circles. Since the temporal sampling rate remained the same at about 20 snapshots per shedding cycle of the time periodic flow, the initial bins contain more snapshots due to the longer period than the time periodic flow bins. The resulting POD mode ensembles were truncated to retain five main modes each (J = 5) and subjected to a second POD decomposition performed on a given main mode from all snapshot ensembles, i.e. on the 11 different modes 1, modes 2, and so on. The results of this second POD decomposition were truncated at either two or three modes for the 5×2 or 5×3 mode model, respectively. The truncated DPOD spatial mode basis was then orthonormalized. Comparing the DPOD based estimation errors for both models (Figure 13), it can be seen that the maximum of the estimation error during the entire start-up transient is reduced to well below one per cent for the 5×2 mode model, and a fraction of one per cent with an almost flat distribution for the entire start-up transient for the 5×3 mode model.



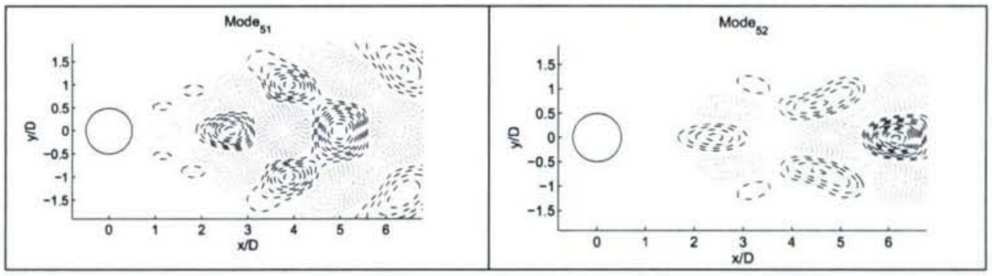


Figure 14: Truncated spatial mode set for transient development of the limit cycle flow field showing 5 × 2 DPOD modes. The first shift modes are in the right column, the main modes in the left. Dashed lines are negative iso-contours of streamwise velocity, solid lines positive iso-contours.

It should be noted that the DPOD models, if not truncated, will recover the exact data set used for their derivation just as the traditional POD models. Inspecting the DPOD spatial modes shown in Figure 14 in more detail, it can be seen that mode i=1, j=1 is the mean flow, followed by modes i=2, j=1 and i=3, j=1 as the Karman vortex shedding modes. These modes together contain more than 90% of the fluctuation energy of the limit cycle. Modes i=4, j=1 and i=5, j=1 are a higher order harmonic mode, with about 4% of the fluctuation kinetic energy in each mode. While not identical, modes i=1, j=1 through i=1, j=5 are very similar to the mean flow and the first four modes of a regular POD decomposition performed on limit cycle data. Mode i=1, j=2, which is the shift mode of the mean flow, is extremely similar and serves the same purpose as the shift mode constructed by Noack et al. (2003) or Siegel et al. (2003). The main improvement of DPOD, however, lies in the shift modes obtained for the main von Kármán vortex shedding and higher modes, modes i=2, j=2 through i=2, j=5. Comparing the minima and maxima of a given fluctuating spatial mode and its shift mode, one can observe that mode and shift mode are out of phase close to the body, and in phase far downstream. Thus, adding a main mode and its shift mode with the same phase will result in weaker peaks close to the cylinder, and stronger peaks farther downstream. This is a situation encountered during the onset of vortex shedding, where the formation length is relatively large. Considering the mode amplitudes of a given fluctuating mode and its shift mode shown in Figure 7, it can be seen that at the beginning of the simulation the amplitudes of the modes and their shift modes are in phase. This correctly models the long formation length observed in inspecting the original data. However, around t/T=9, the shift mode amplitudes experience a phase reversal with respect to their respective main mode amplitudes, and from then on remain 180 degrees out of phase. This effectively results in a subtraction of the spatial mode and its corresponding shift mode, which results in stronger modal peaks close to the cylinder, and weaker peaks further downstream. This correctly models the flow state encountered during the limit cycle, where the formation length is shorter and the vortices form closer to the cylinder. We conclude from these observations that the shift modes obtained from the DPOD procedure achieve the same purpose for adjusting the formation length as the mean flow shift mode does for the mean flow: It allows the POD model to correctly capture the changes encountered during transient flow situations.

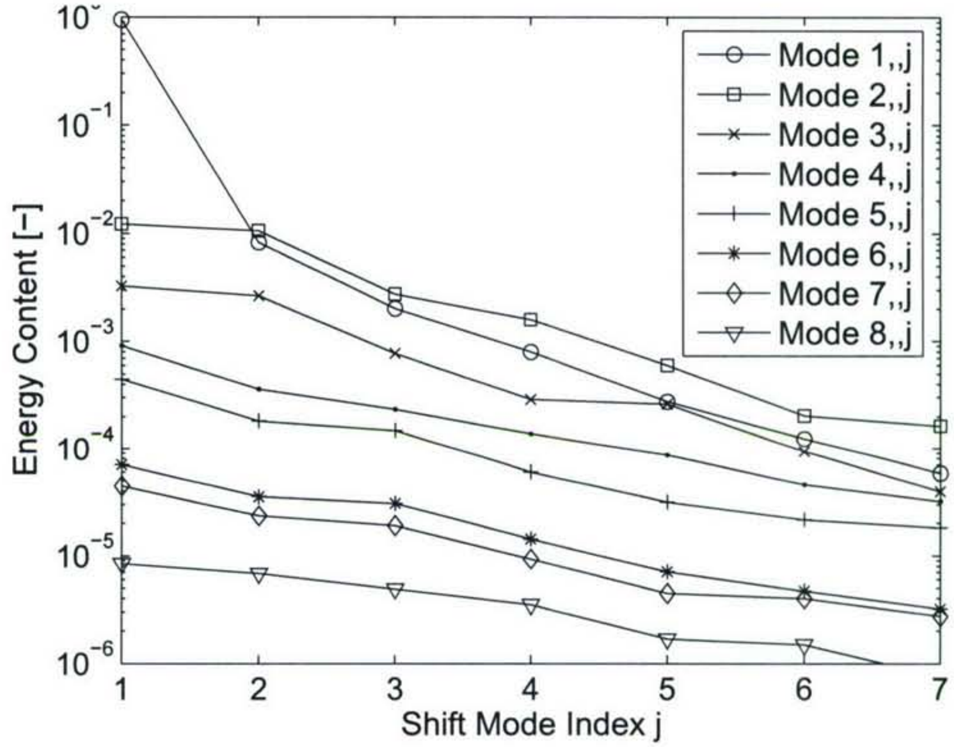


Figure 15: Energy Content for the DPOD model developed from transient Reynolds number change from Re = 40 to Re = 100 data shown in Figure 2, as well as transient open loop forced data.

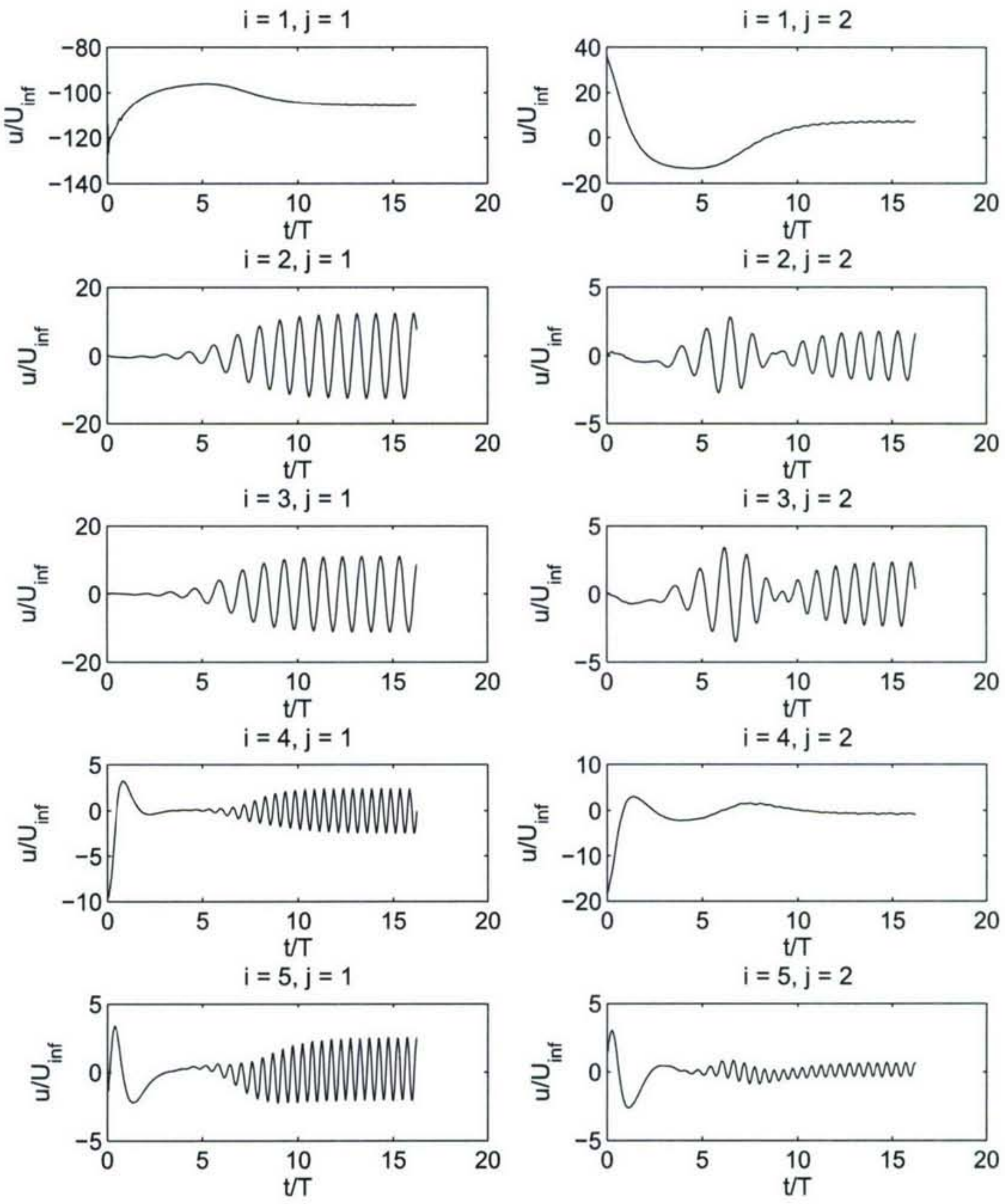


Figure 16: Mode amplitudes a_{ij} of the first 5 × 2 DPOD modes during transient development of the unforced cylinder wake limit cycle, Re = 100. For corresponding spatial modes, refer to Figure 14.

4.1.3 Transient Forced Simulations

With these encouraging results in modelling unforced flow behaviour, we proceed to examine the effectiveness of the DPOD procedure in modelling the transient behaviour of open loop forced flow fields. The effect of cylinder translation on the vortex shedding behind a circular cylinder has been investigated in literature in detail, starting with the investigations of Koopmann (1967). The parameter space is two-dimensional considering both the non-dimensional peak cylinder displacement, A/D, and the forcing frequency normalized by the natural shedding frequency f_0, $(f-f_0)/f_0$. Figure 17 shows the results of Koopmann (1967) indicating the so called lock-in region, within which the vortex shedding exhibits a fixed phase relationship with the forcing. Outside of this lock-in region, the flow response to the forcing is chaotic. The question is now how well (if at all) a DPOD model can capture the wake behaviour within this lock-in region, given only a limited number of simulations to develop the model. Of particular interest is the question if the DPOD model will yield usable flow field estimates for data sets that were not used for the development of the spatial modes. Note that in the transient start-up model development presented above, the flow field used for model development and for state estimation were identical.

For the transient forced flow field investigations, we employ four simulation data sets obtained for different forcing conditions, indicated in Figure 17, to derive a set of spatial DPOD modes. We then proceed to project the snapshot data onto these spatial modes to derive the mode amplitudes not just for the data sets from which they were obtained, but also for four additional data sets within the lock-in region that were not used for the spatial mode development. The rationale behind this is to test the robustness of the model for off-design conditions: Even within a parameter space that is only two-dimensional, it is impossible to include all possible combinations of forcing parameters in the model development. Thus, it is of interest to know how much information is needed to yield a stable and accurate flow model for an entire range of forcing conditions, in this case, for the lock-in region at small amplitudes.

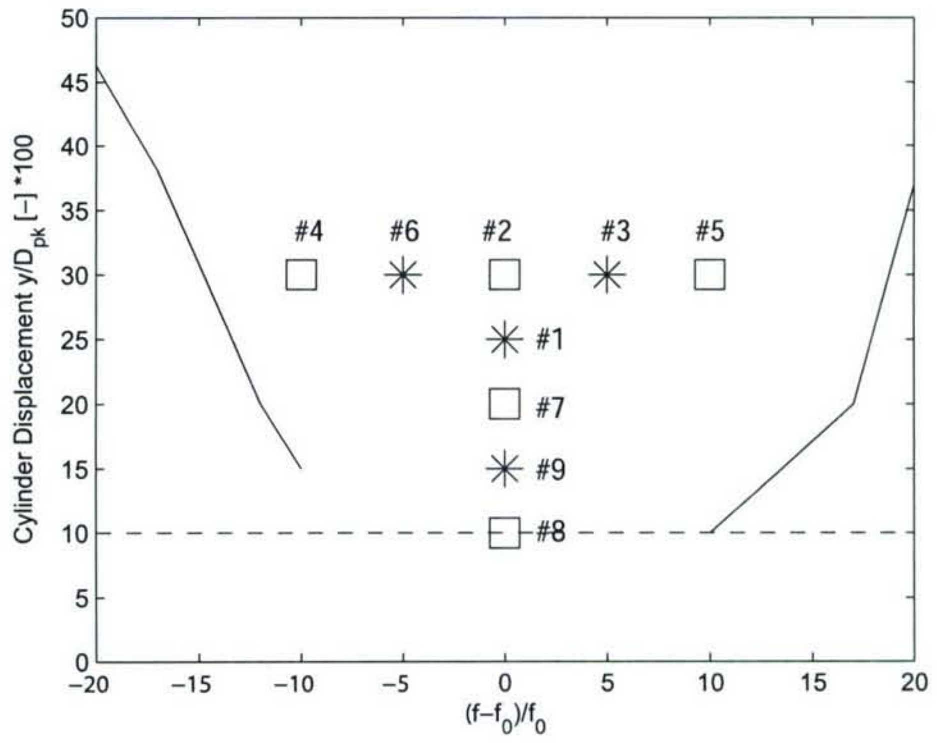


Figure 17: CFD simulation data sets used for mode set derivation and mode set verification. Solid lines indicate the limits of the Lock-In boundary as identified by Koopman (1967). Data sets identified by a square are used for spatial mode derivation. Data sets identified by stars are used for mode verification. The case numbers are referenced in the text and in Table 3

The simulations for obtaining open loop forced data were restarted with the flow solution from the end of the unforced limit cycle simulation presented above. In order to prevent non-physical transients in the flow, the cylinder displacement was prescribed as a C^2 continuous function $y_c(t)$. The forcing was started exactly out of phase with the vortex shedding, which yields the longest possible transient while the phase of the shedding adjusts to the phase of the forcing, and was applied for 15 periods of the forcing frequency. A typical forcing signal is shown in Figure 18a. The flow response to the forcing is dependent on both the amplitude and the frequency of the forcing; in general, the smaller the forcing amplitude, the longer the transient after activation of the forcing. Figure 18b shows the transient for the particular forcing conditions to last for about 10 shedding cycles before a stable phase relationship between forcing and vortex shedding is established. The unsteady lift coefficient in this phase locked state is larger than in the unforced flow field, indicating stronger vortex shedding, which results in an increase in the drag coefficient c_D as well. While shown for one forcing condition only, all open loop forced cases investigated show this behaviour, even though the amount of increase in lift fluctuations and drag coefficient varies for different forcing parameters. After the forcing is stopped, the flow resumes its original shedding pattern, indicated by a return of lift fluctuations and drag coefficient to the unforced values. The amount of time that this second transient lasts again depends on the forcing parameters. Both transient phases yield important information on the dynamics of the flow and its interaction with the forcing.

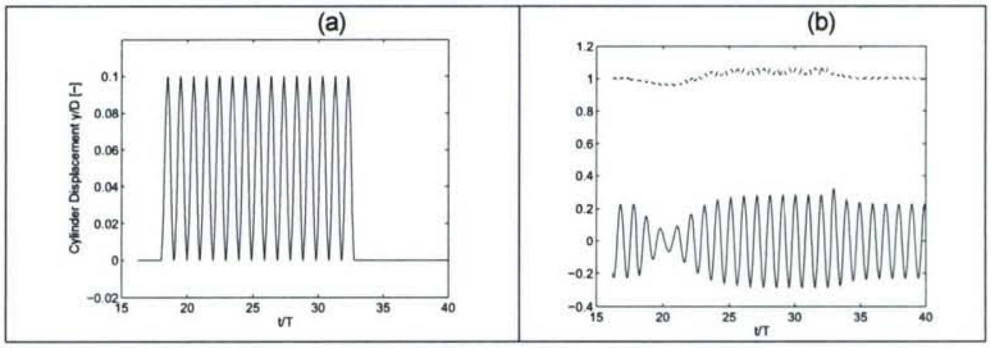


Figure 18: (a) Cylinder displacement. $f/f_0 = 1$; $A/D = 0.05$. Forcing activated at $t/T = 18$ and stopped at $t/T = 33$, after 15 full forcing cycles. (b) Transient forced normalized lift coefficient c_l/c_{d0} (solid line) and drag coefficients c_d/c_{d0} (dashed line).

The goal of developing a low dimensional model is to capture the flow behaviour in these situations accurately, not just for the unforced and periodically forced flow states. The first step towards this goal is to develop a spatial mode set that is able to represent the flow state correctly during all phases, unforced, forced as well as transient. In order to achieve this, DPOD was applied to the four simulation runs indicated in Figure 17 just as for the unforced start-up simulation presented above. The DPOD decomposition was truncated at 3 main modes (consisting of the mean flow and the von Karman modes) and one shift mode, yielding the total DPOD mode set of size 3x2 modes shown in Figure 19. Comparing the mean flow and its shift mode obtained from the transient forced simulations to those obtained during the start-up of the unforced flow shown in Figure 14, it appears that the roles of mode and shift mode are reversed, but otherwise the spatial distribution is similar. This is to be expected, since the majority of the data in the transient forced simulations is acquired with the forcing active. Thus, more energy is contained in the modes modelling the controlled flow, making them dominant in terms of energy over the modes representing the unforced flow. The mode amplitudes shown in Figure 20 confirm this: The amplitude of, for example, mode i=2, j=1 is larger than the amplitude of mode i=2, j=2 during the time when the forcing is active.

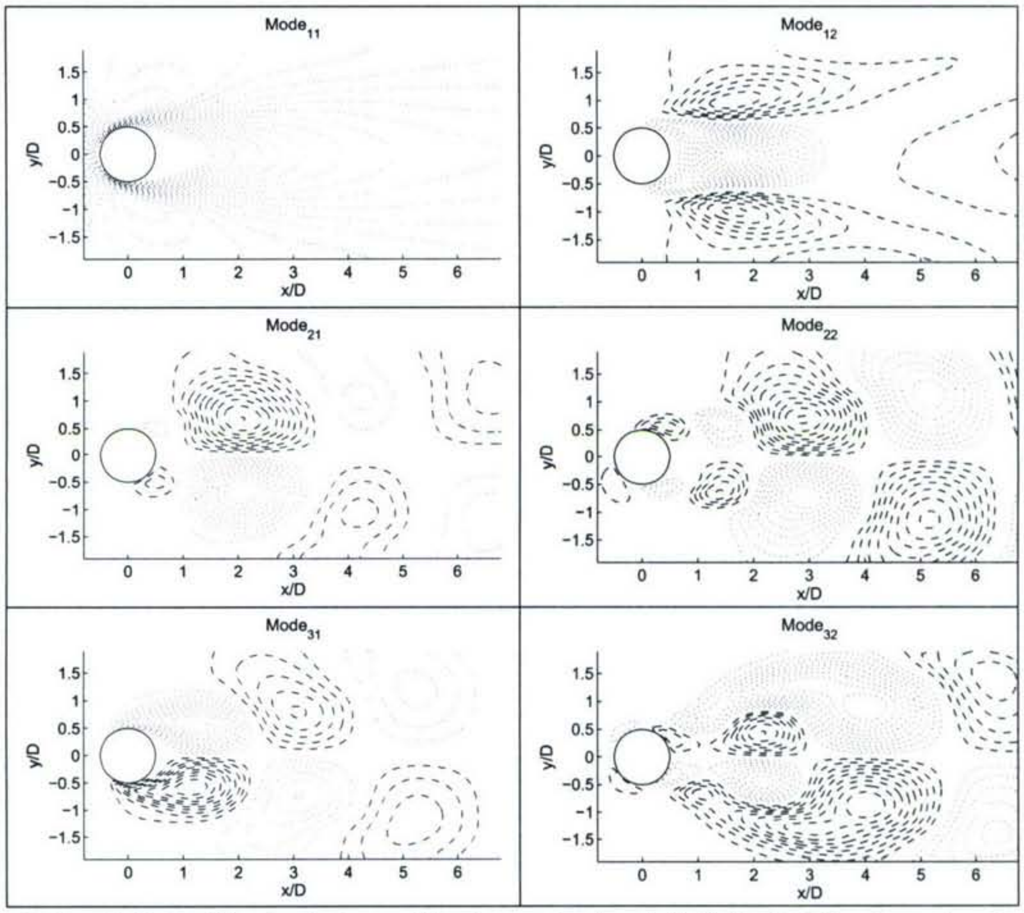


Figure 19: Transient forced DPOD spatial mode set using one shift mode for each main mode, the first 3 x 2 DPOD modes are shown. Iso-contours of streamwise velocity are shown, solid lines are positive, dashed lines negative.

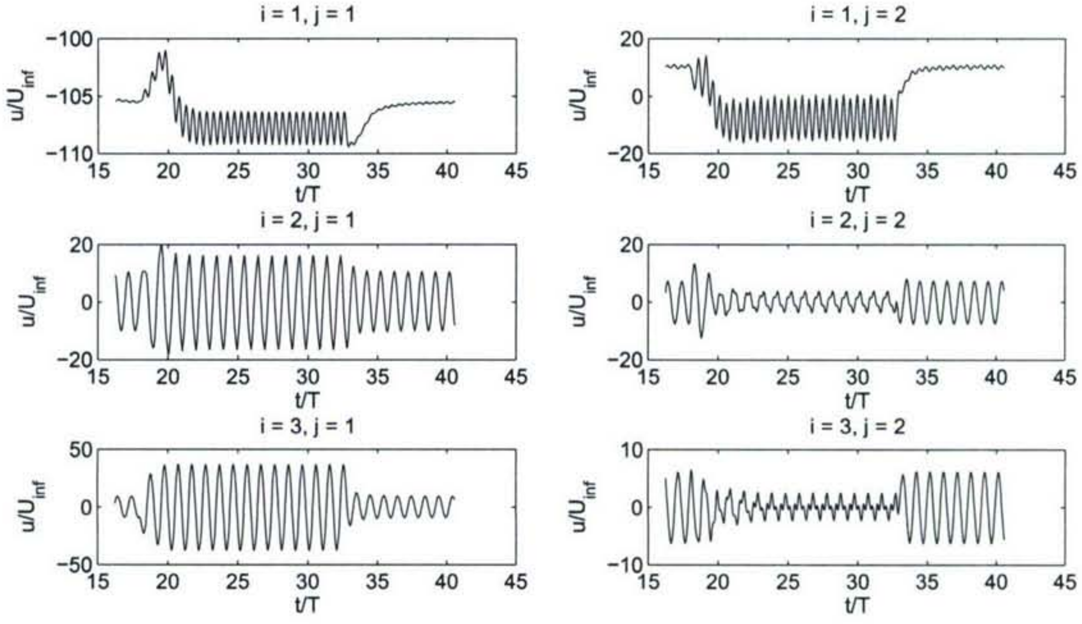


Figure 20: Mode Amplitudes a_{ij} of the open loop forced simulation $f/f_0 = 1$ and $A/D = 0.25$. Forcing activated at $t/T = 18$ and stopped at $t/T = 33$, after 15 full forcing cycles.The first 3 x 2 DPOD mode amplitudes are shown.

In order to minimize the number of modes as much as possible, the DPOD model was truncated to include only main modes $i \leq 3$, which contains the mean flow and the two von Kármán modes. This 3x2 mode model was then projected onto all eight of the open loop forced simulations performed within the lock-in region indicated in Figure 17, including the four simulations used for derivation and the additional simulations performed for model verification only. The mode amplitudes obtained by this projection for one of the simulations that was *not* used to derive the spatial modes is shown in Figure 20. These mode amplitudes were then used to reconstruct the flow field, and the reconstructed flow field was compared to the simulation results. The modelled flow field, reconstructed from these mode amplitudes and their respective spatial modes, was then compared to the results of the numerical simulation. The average error within the entire x/D, y/D POD plane is shown in Figure 21. While the error is about one percent of the free stream velocity during the unforced times ($t/T<18$ and $t/T>33$) at the beginning and end of the simulation, it is largest during the transient encountered after starting the forcing, with a second smaller peak when the forcing is turned off. During the phase-locked portion of the simulation ($t/T = 22\ldots33$) the error is about 4 percent of the free stream velocity, or four times larger than in the unforced case.

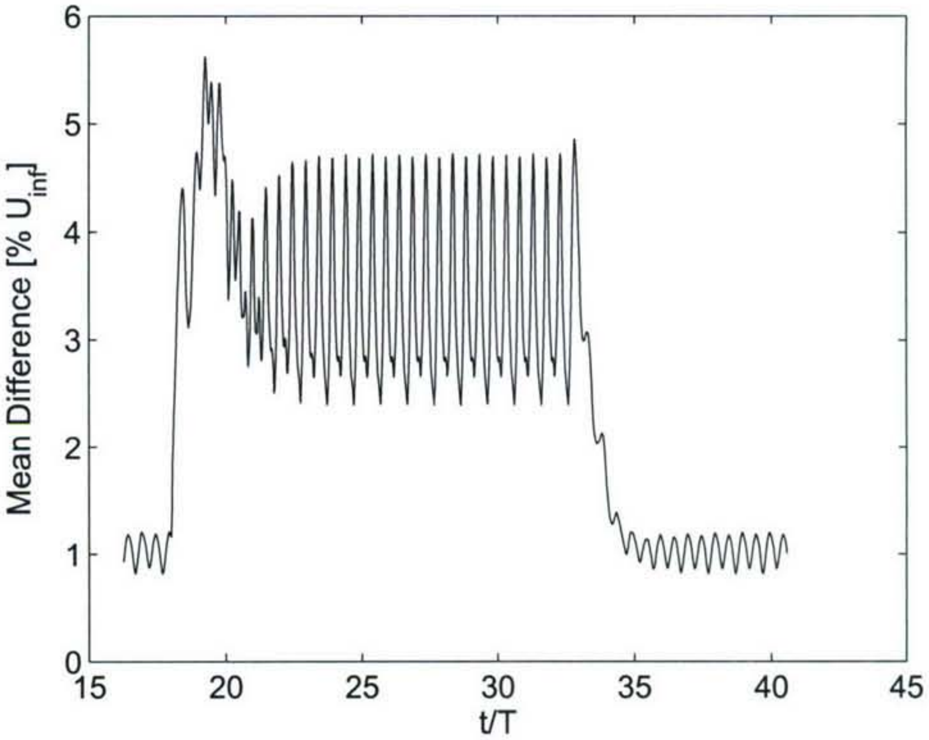


Figure 21: Estimation error of the reconstructed flow field compared to the actual flow field. Simulation $f/f_0 = 1$ and A/D = 0.25. Forcing activated at t/T = 18 and stopped at t/T = 33, after 15 full forcing cycles. Mode set of 3 x 2 DPOD modes.

There are two main reasons for this increase: First, since this particular data set was not used for the development of the spatial modes, the match of the spatial modes to the data is less than optimal. And second, the open loop forcing creates larger amplitudes of higher frequency disturbances, as can be seen by inspecting the fluctuation amplitudes in the error during the unforced and locked-in portions of the simulation. During the lock-in portion, these fluctuations are significantly larger, indicating more energy in the higher order modes. Since these modes have been truncated, however, the error increases due to spill-over effects. Overall, however, the modelling error never exceeds 6% of the free stream velocity, and for most instances of time is well below five percent. This is more than sufficient for typical feedback control purposes. Furthermore, the error may be reduced by retaining more modes using less aggressive truncation of the model. This is shown in Table 1, where the same DPOD model is truncated to different numbers of main and shift modes. By doing an actual comparison between the original CFD data and the truncated DPOD estimates, it can be seen that the estimation error does get smaller as the number of modes retained in the model is increased. This decrease in error is consistent with the roll off seen in the mode energy plot, Figure 15. While only one particular data set, the forcing using 25% of the cylinder diameter displacement at the natural shedding frequency, is presented for reasons of brevity, all investigated data sets show errors over time similar to that presented in Figure 21.

	Truncation 3x2	Truncation 5x3	Truncation 5x5	Truncation 8x7
Main Modes	3	5	5	8
Shift Modes	2	3	5	7

Number of Modes	10	15	25	56
Estimation Error [% U_∞]	3.86	1.67	1.03	0.36

Table 1: L2 norm estimation error of the streamwise velocity component in percent of freestream velocity averaged over the entire flow field and time as a function of mode truncation. The data set analyzed is the transient development of the limit cycle.

The time averaged errors for all different data sets are summarized in Table 2. These results are based on a 6 mode model consisting of modes 11 through 32 shown in Figure 19. These same modes were projected onto all data sets in the same fashion as presented for the forcing case A/D=0.25, f/f_0 = 1 in order to determine the errors shown in Table 2. While it is expected to find larger errors for data sets not used for spatial mode development, the error magnitude correlates more with forcing amplitude than anything else. This is another indication that modal truncation drives the error: For larger forcing amplitudes, more energy is contained in the truncated higher order modes. With the largest error at 3.18 per cent, however, the agreement between the model and the flow field can be considered to be quite good. This agreement is a necessary prerequisite for the development of a low dimensional mathematical model based on the mode amplitudes which will be described in the following chapter.

Case Number	Forcing Frequency (*100 % f_0)	Forcing Amplitude (y/D)	Time Averaged Mean Flow Field Difference (%U_{inf})
1	1	0.25	2.65
2	1	0.3	3.00
3	1.05	0.3	2.97
4	0.9	0.3	2.97
5	1.1	0.3	3.18
6	0.95	0.3	2.91
7	1	0.2	2.38
8	1	0.1	1.79
9	1	0.15	2.09

Table 2: Average estimation error for all different transient forced simulations using the DPOD modes truncated to 5x3 modes.

In summary, DPOD is a computational tool that is suitable to develop low dimensional POD bases that span a variety of flow conditions, both from different Reynolds numbers and different forcing inputs. Additionally, transient flow situations can be modelled with good accuracy as well. The resulting DPOD basis is also valid for flow situations that were not included in its derivation, as long as they are within the parameter range spanned by the snapshots used for derivation of the basis, which constitutes an interpolation capability of the DPOD basis.

4.1.4 Dynamic Model for transient and feedback controlled flow

With the DPOD spatial mode basis, developed in the previous section, covering a range of both Reynolds numbers and forcing conditions, the entire time dependent global dynamic behaviour of the flow is captured in the corresponding mode amplitudes. Thus, the next goal is to develop a set of equations describing the dynamic behaviour of these mode amplitudes. These equations are needed both for development of control algorithms, as well as for testing of these controllers. Traditionally, Galerkin projections of various types have been used to project the mode amplitudes onto the Navier Stokes equations. However, this approach has led to a variety of problems which are discussed both in the introduction and the following section, leading us to use of a different modelling approach as described in the following section.

The nonlinear ANN-ARX model was developed using a single design case (Case #1) for training of the ANN within the lock-in region. Eight simulations not used for model development (#2 - #9), shown in Figure 17, were used for validation of the model. The RMS error in per cent of the mode amplitude, for the 5 × 3 modes, is presented in Table 3. Each case is comprised of 490 time steps. As seen in Table 3, the validation of the ANN for eight different cases within the lock-in region provided very promising results and shows accuracy as well as stability and robustness.

Mode i,j	Case #1	Case #2	Case #3	Case #4	Case #5	Case #6	Case #7	Case #8	Case #9
1,1	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04
2,1	0.27	0.25	0.24	0.30	0.25	0.27	0.29	0.37	0.45
3,1	0.25	0.26	0.26	0.22	0.30	0.23	0.25	0.29	0.33
4,1	0.77	0.69	0.67	0.83	0.70	0.74	0.88	1.18	1.18
5,1	1.79	1.72	1.74	1.73	1.88	1.74	1.89	2.28	2.65
1,2	0.52	0.45	0.42	0.54	0.46	0.47	0.67	1.23	1.28
2,2	1.13	1.00	0.99	1.02	0.95	1.01	1.34	2.43	3.30
3,2	0.46	0.42	0.38	0.45	0.37	0.40	0.57	1.25	2.06
4,2	0.97	0.90	0.80	0.89	0.72	0.82	1.16	1.97	2.40
5,2	1.60	1.11	1.32	1.34	1.10	1.52	1.86	2.48	3.72
1,3	28.61	1.71	32.49	25.90	31.96	29.50	24.07	12.62	6.54
2,3	19.40	0.96	21.07	15.03	19.20	18.43	17.83	15.07	12.78
3,3	19.76	1.26	16.16	26.45	13.73	25.39	19.21	13.94	9.14
4,3	16.97	1.89	18.01	20.06	18.21	19.62	15.67	11.65	7.93
5,3	16.34	2.07	21.53	16.76	28.15	17.42	16.27	15.03	10.31

Table 3: RMS errors in per cent of the mean mode amplitude a_{ij} for DPOD-ANN-ARX estimated mode amplitudes compared to the DPOD mode amplitudes obtained from CFD simulation. The DPOD model is truncated to 5 × 3 modes for both model and CFD results

Figure 22 and Figure 23 show the CFD based DPOD mode amplitudes in comparison to the ANN-ARX model estimates for design case (#1) (Figure 22) and one of the off-design cases (#6) (Figure 23). Note the accurate estimation of the mode amplitudes by the ANN-ARX model when the forcing is switched on and off in a transient fashion. While we present only one off-design case, the other cases are qualitatively and quantitatively similar to case #6 presented in Figure 23. As reported by Cohen et al. (2006), attempts made at modelling these transients with a linear system identification approach were not nearly as successful.

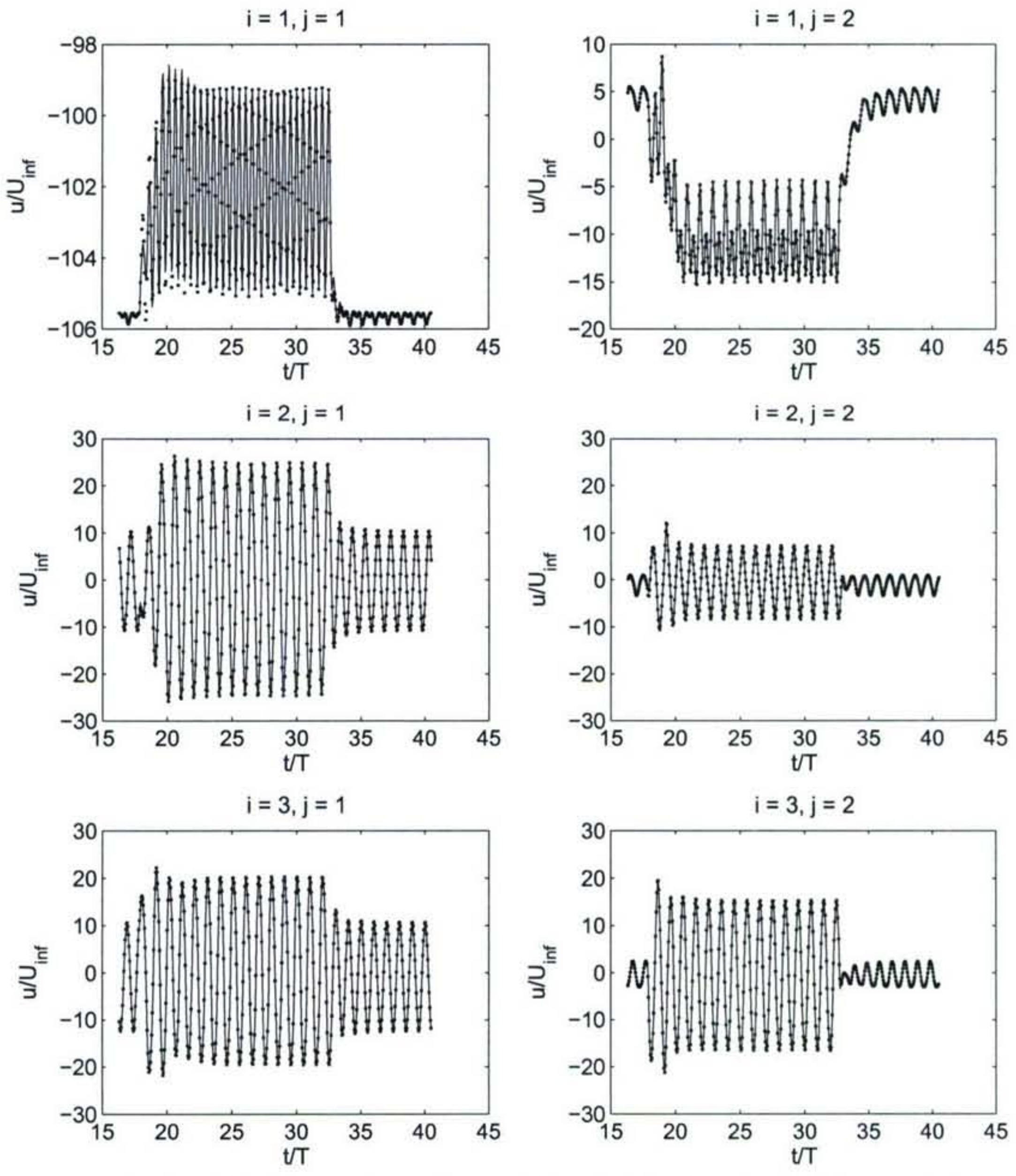


Figure 22: Mode amplitudes a_{ij} of the first 3 x 2 DPOD modes for the design case, forcing with $f/f_0 = 1$ and $A/D = 0.25$. This data was used for training of the ANN-ARX network. Lines show mode amplitudes from the CFD simulation. Symbols (•) represent the mode amplitude estimation from the ANN-ARX model.

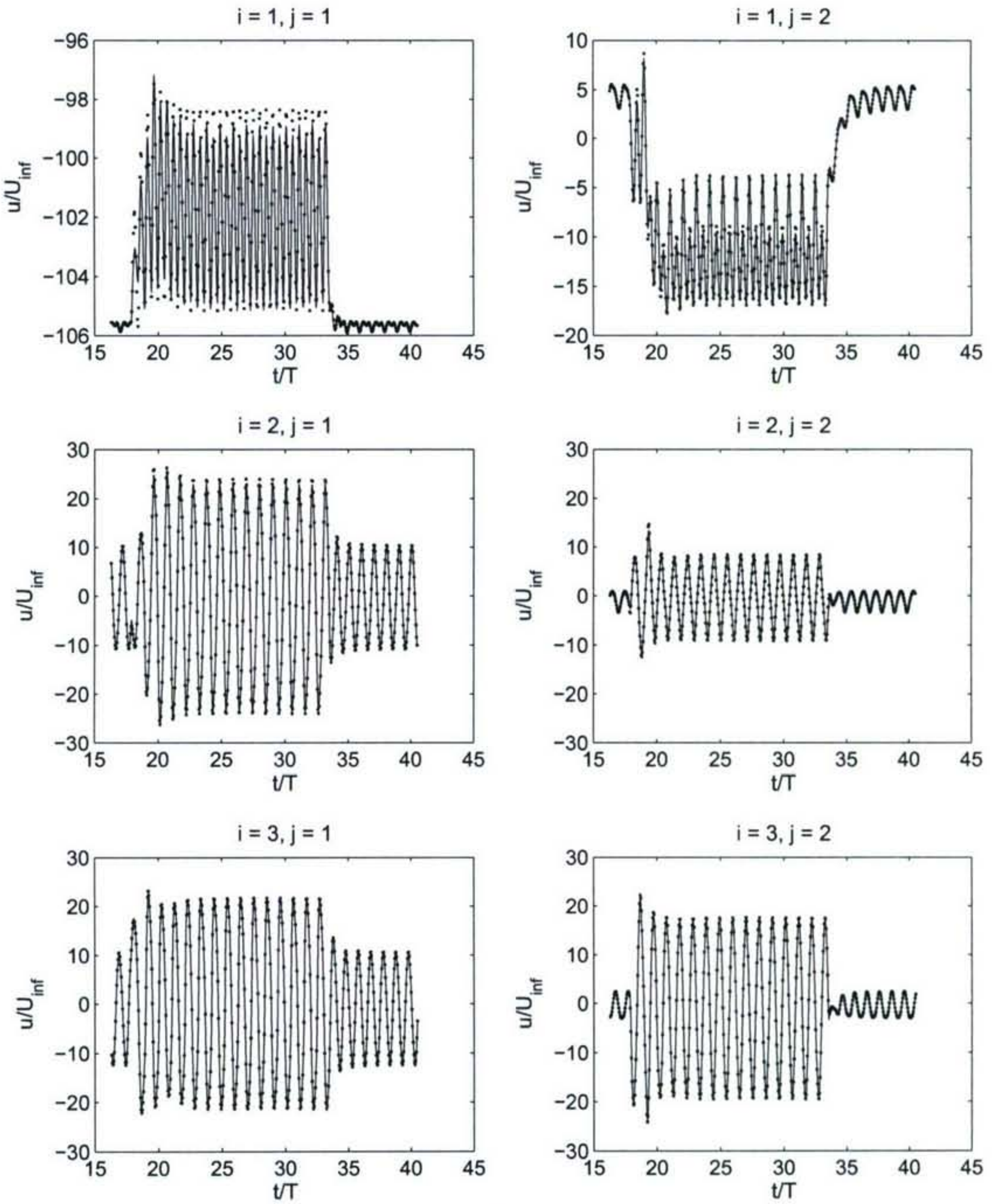


Figure 23: Mode amplitudes a_{ij} of the first 6 DPOD modes for the off-design case #6, forcing with $f/f_0 = 0.95$ and $A/D = 0.30$. This data was not used for training of the ANN-ARX network. Lines show mode amplitudes from the CFD simulation. Symbols (•) represent the mode amplitude estimation from the ANN-ARX model.

Another important model property is robustness against changes in operating conditions, in our case, the Reynolds number. While the main goal of model development and training was to obtain a model that accurately represents the flow changes due to forcing, it is important that the model delivers accurate results in the vicinity of the design point with respect to Reynolds number. We thus ran unforced (actuation amplitude A/D = 0) ANN-ARX simulations at different Reynolds numbers between 60 and 100. These are presented in Figure 24 in comparison to a curve fit to experimental data presented by Williamson (1996). While not shown, all of the simulations at different Reynolds numbers developed stable limit cycles with no divergence of the mode amplitudes over time, demonstrating the long term stability of the model. The ANN-ARX results show good agreement with experimental data, even though training of the model was by no means optimized to obtain accurate results at different Reynolds numbers. This could be improved upon by adding more transient Reynolds number data to the training data ensemble. Also shown in comparison are Galerkin-model results presented by Noack et al. (2003), obtained with the best published model for the circular cylinder wake at a Reynolds number of 100, according to the authors' best knowledge. Their models A and B, which use eight POD modes and eight POD modes plus a shift mode, respectively, were obtained using a Galerkin projection of the mode amplitudes onto the Navier Stokes equations. Model A is equivalent to the benchmark model by Deane et al. (1991), while Model B includes the shift mode developed from the steady solution of the flow field. It can be seen that the ability of both models A and B to correctly capture the natural shedding frequency at off-design Reynolds numbers falls short of the performance of the ANN-ARX-DPOD model developed in this work.

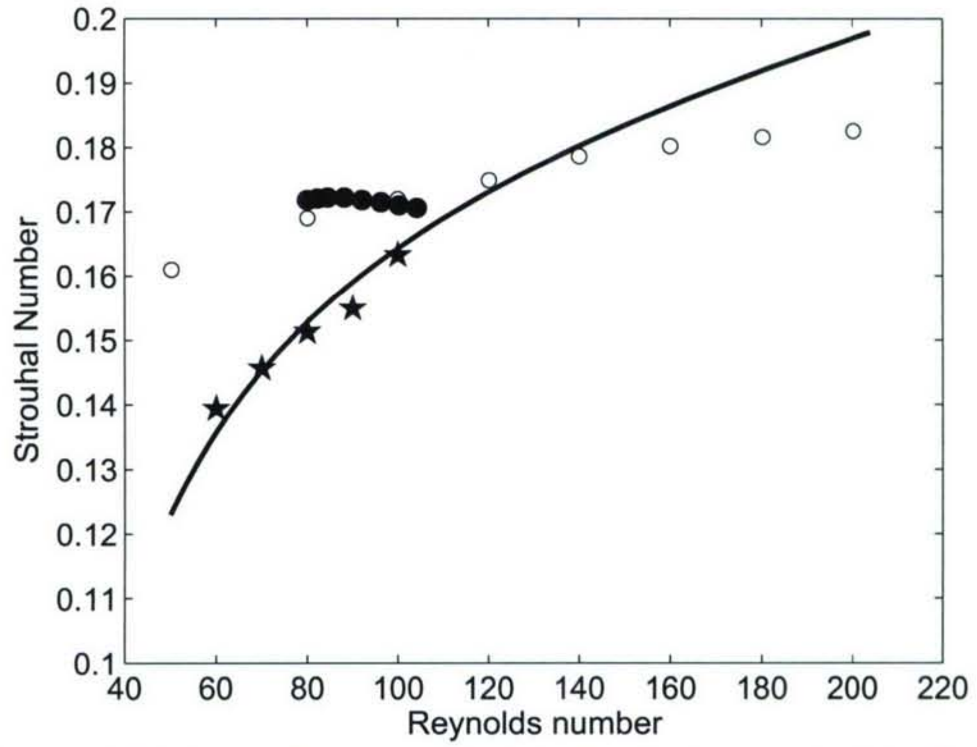


Figure 24: Comparison of ANN-ARX model of unforced vortex shedding at different Reynolds numbers to experimental data and models developed by Noack et al. (2003). Solid line, curve fit of experimental data described by St = -3.3265/Re+0.1816 +1.6e-4 Re presented in Williamson (1996). (*), ANN-ARX model.(.), Model A presented in Noack et al. Fig. 12a. (o), Model B presented in Noack et al. Fig. 12a.

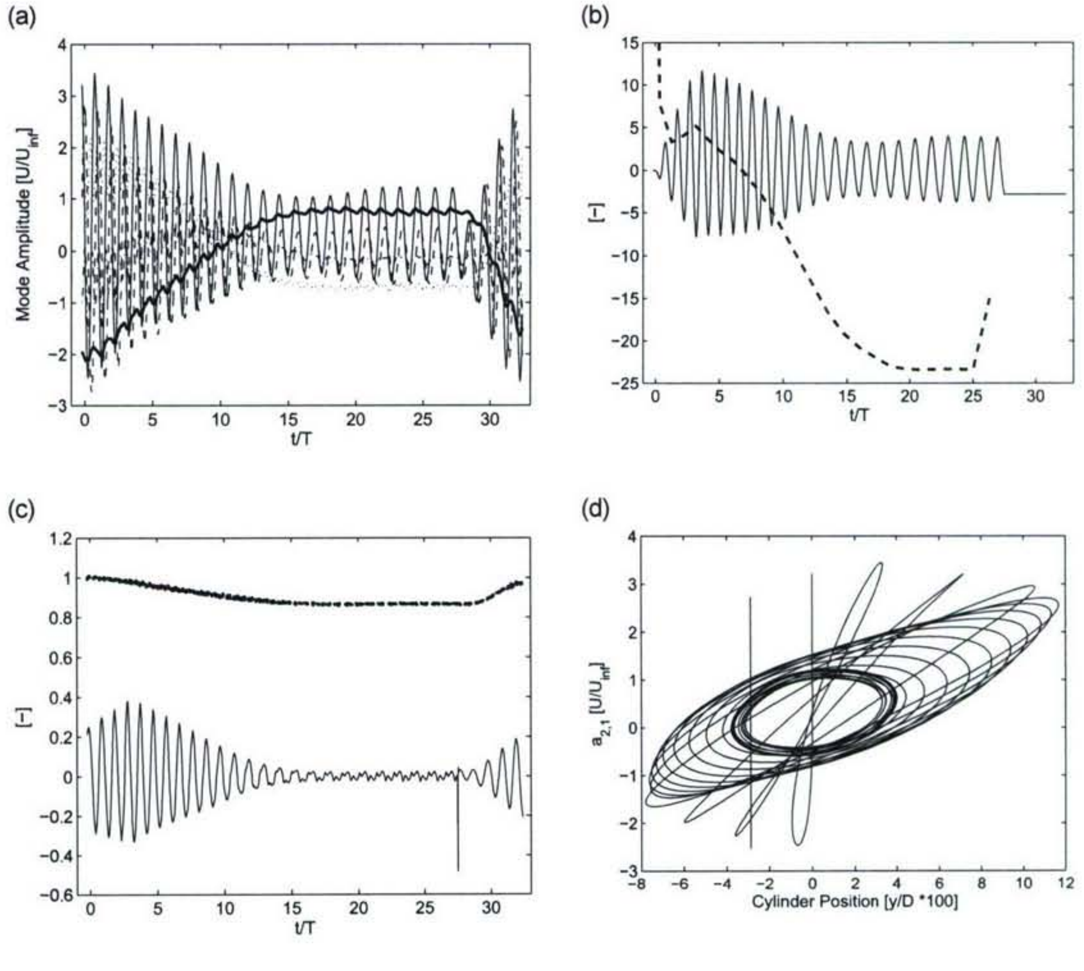
In addition to the open loop forced validation cases and the off-design Reynolds number studies presented above, a closed loop simulation result was used to test the DPOD-ANN-ARX model's ability to estimate feedback controlled cylinder wake flows. These results have been presented earlier by Siegel, Cohen & McLaughlin (2006). Figure 25 shows the pertinent results of this CFD simulation in terms of mode amplitudes, lift, drag and controller phase. The feedback control law was a PD controller with adjustable gains. For all investigations, only displacement of the cylinder in the flow normal direction was employed for forcing the flow. The control algorithm acts on amplitude estimates of POD modes, where the POD model in this investigation included four main modes and a shift mode. These modes are qualitatively similar to modes i=1, j=1 – i=4, j=1 and the shift mode is similar to mode i=1, j=2 as shown in Figure 10. This design decision was made based on our earlier investigations controlling a low dimensional model of the flow (Cohen et al. 2003). For the low dimensional model, proportional gain applied to the first von Kármán mode was sufficient to suppress vortex shedding. The employed PD feedback control strategy can be written as

$$y_{cyl} = K_p \cdot a_{21} + K_d \cdot \frac{da_{21}}{dt}. \tag{17}$$

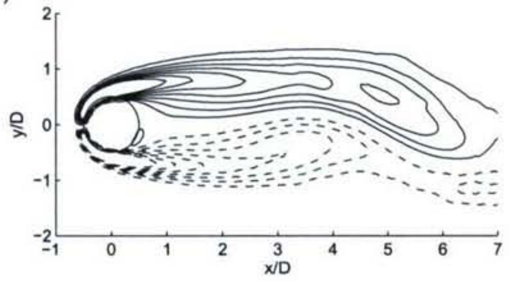
Instead of directly specifying the K_p and K_d gains, they can be expressed in terms of an overall gain K and a phase advance φ,

$$K_p = K \cdot \cos(\varphi)$$
$$K_d = \frac{K \cdot \sin(\varphi)}{2\pi f}, \tag{18}$$

where f is the natural vortex shedding frequency.



(e)



(f)

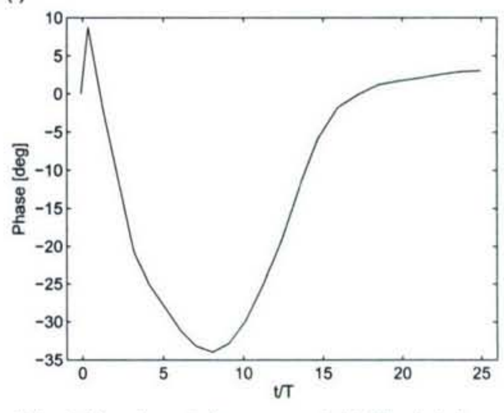


Figure 25: Linear feedback of mode amplitude a_{21} with a linear proportional-differential variable gain controller as presented in Siegel et al. (2006). K_0=1e-3, φ_0 = 15 deg, K_A=1e-6, K_φ=0.75 deg. (a) Mode amplitudes; a_{21} (-), a_{31} (--), a_{41} (.), a_{51} (.-),a_{12} (•) (b) Cylinder displacement y/D*100 (-) and frequency f-f₀/f₀(--), (c) Nondimensional lift (thin line) L/D₀ and drag (thick line) D/D₀ forces. (d) Phase between Cylinder Position y and a_{21}, (e)Instantaneous vorticity contours at t/T=24.7, (f) Phase advance during the run. The controller is activated at t/T = 0 and deactivated at t/T = 27.5

During the course of the investigations, we found it advantageous to implement a variable gain strategy. The basic idea is that as the flow field is modified from its original state, different values for the phase advance φ and the gain K may be more advantageous than those effective in controlling the unforced flow field. Using the amplitude difference between the unforced state and the current state, K and φ are written as

$$\begin{aligned} K &= K_0 + K_A\left(a_{12} - a_{12}\big|_{t=0}\right) \\ \varphi &= \varphi_0 + K_\varphi\left(a_{12} - a_{12}\big|_{t=0}\right) \end{aligned} . \tag{19}$$

Equation 6.3 details the adjustment scheme for the variable gain strategy. Since mode 12 is a good measure of the change in the mean flow, the feedback gain K and the phase advance φ are adjusted from their initial values K_0 and φ_0 in proportion to the change in mode i=1, j=2 by applying a phase advance factor K_φ and a gain change factor K_A. These additional factors can be either applied together or individually.

Figure 25, reproduced from Siegel et al. (2006), shows that a drag reduction of about 15% of the total drag was achieved, as well as a lowering of the unsteady lift force by more than 90%. The controller employed by Siegel et al. (2006) was empirical in nature, and while model based estimation of the flow state was a crucial enabling technology, the adjustment of the linear proportional and differential feedback gains was based on trial and error rather than on a model based flow analysis. Nonetheless, this simulation provides an excellent test case to investigate the use of the low dimensional model developed in this work for controller development and testing. We therefore applied the control input that was used in the CFD simulation to the low dimensional DPOD-ANN-ARX model described above. Figure 26 and Figure 27 show the CFD results in comparison to the DPOD-ANN-ARX model results for the 5x3 DPOD mode amplitudes. Although only open-loop and transient startup data was used for model development, the model based estimation of the impact of feedback control on the flow of this closed-loop simulation is fairly accurate. This result demonstrates that the DPOD-ANN-ARX model captures the flow physics correctly, enabling both systematic controller development and performance evaluation.

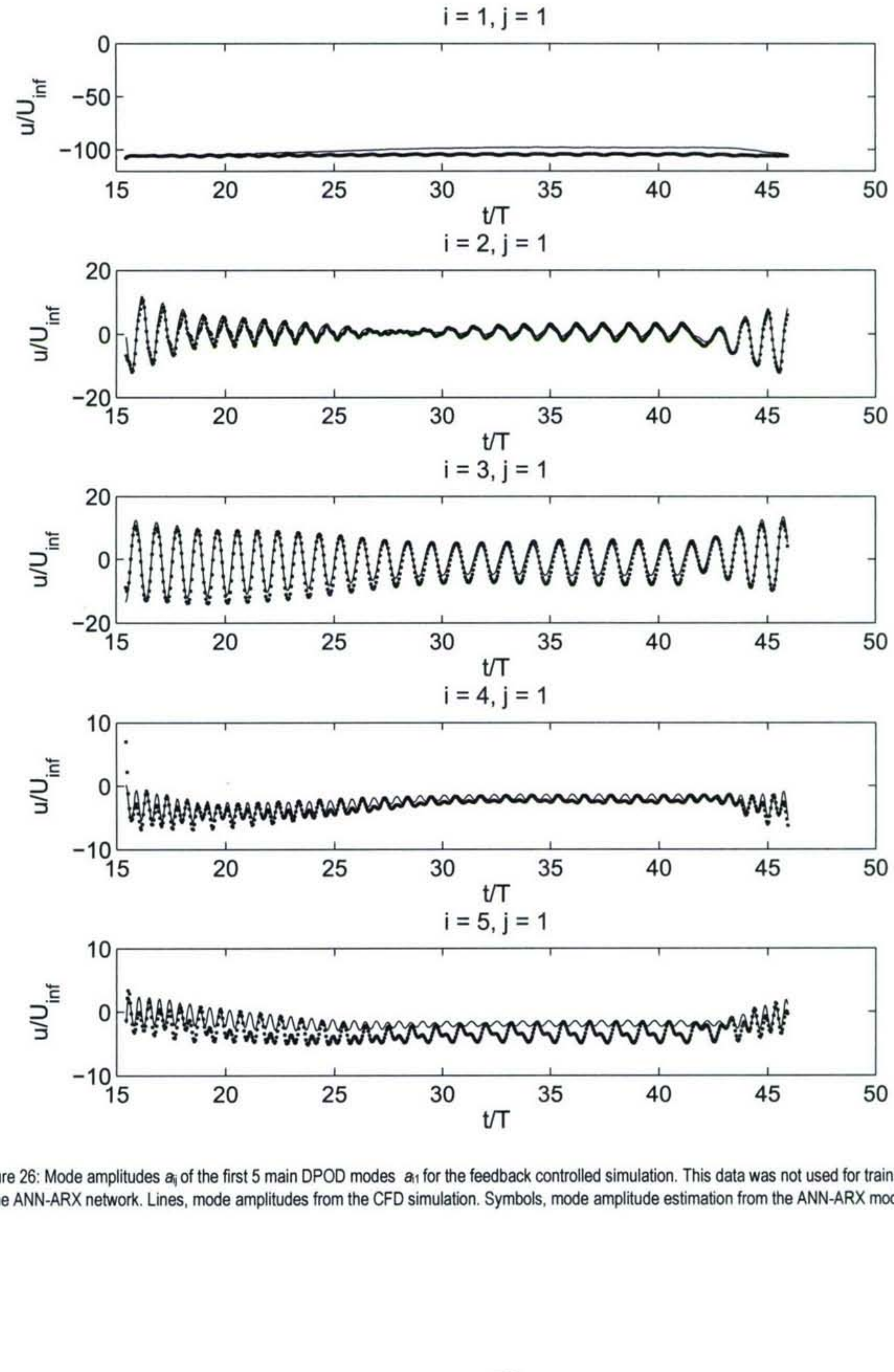


Figure 26: Mode amplitudes a_{ij} of the first 5 main DPOD modes a_{i1} for the feedback controlled simulation. This data was not used for training of the ANN-ARX network. Lines, mode amplitudes from the CFD simulation. Symbols, mode amplitude estimation from the ANN-ARX model.

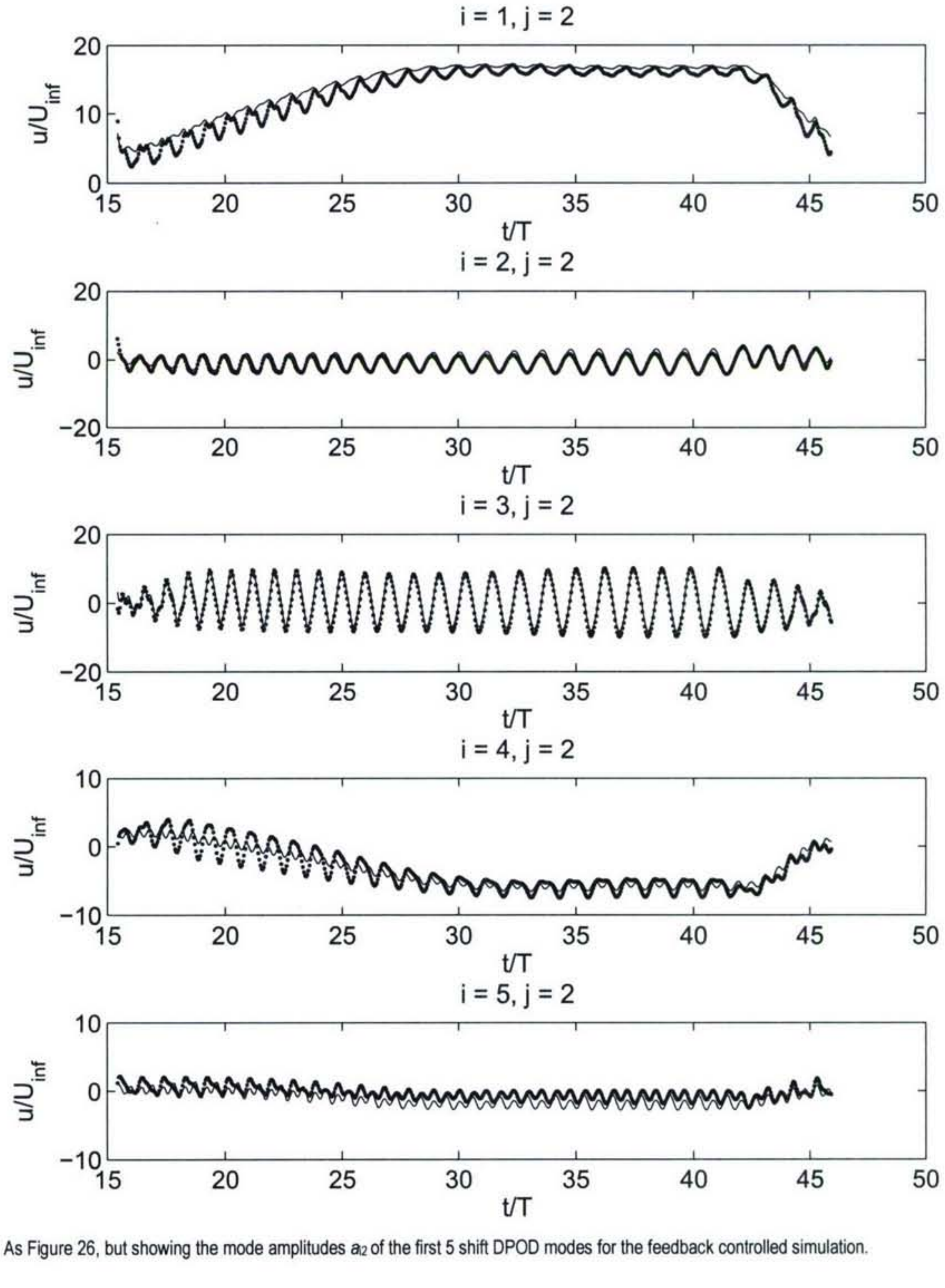


Figure 27: As Figure 26, but showing the mode amplitudes a_{i2} of the first 5 shift DPOD modes for the feedback controlled simulation.

4.1.5 SISO and MISO results

Based on the detailed study of the Low Order Model presented above, this section presents the results both of single mode feedback, as well as multi mode feedback based on the DPOD model. The effectiveness of the single mode feedback was evaluated by keeping the overall gain K constant at K=5e-4, while varying the phase φ of the feedback from 0 to 360 degrees. The controller amplitude was chosen such that the overall cylinder displacement would remain within the range of validity of model, i.e. limiting the maximum displacement $y/D < 0.3$. The resulting closed loop lift and drag force are shown in Figure 28. It can be seen that for the range of feedback phases between about 30 and 250 degrees the drag is increased compared to the unforced flow field, while between 250 and 30 degrees a decrease in drag is observed. The normalized lift and drag forces for both $\varphi=90^\circ$ (drag increase) and $\varphi=330^\circ$ (drag decrease) is shown in Figure 29. In both cases the wake is stabilized at its new state, with about 10% drag decrease for the feedback phase of 330 degrees. The DPOD mode amplitudes for this case are shown in Figure 30. The amplitude of the mode used for feedback, M21, is greatly reduced by the effect of feedback control. However, the amplitude of its first shift mode, M22, is actually increased as a result of feedback. This observation led to the implementation of multi mode feedback, where both M21 and M22 are fed back with individual gains K and phases φ.

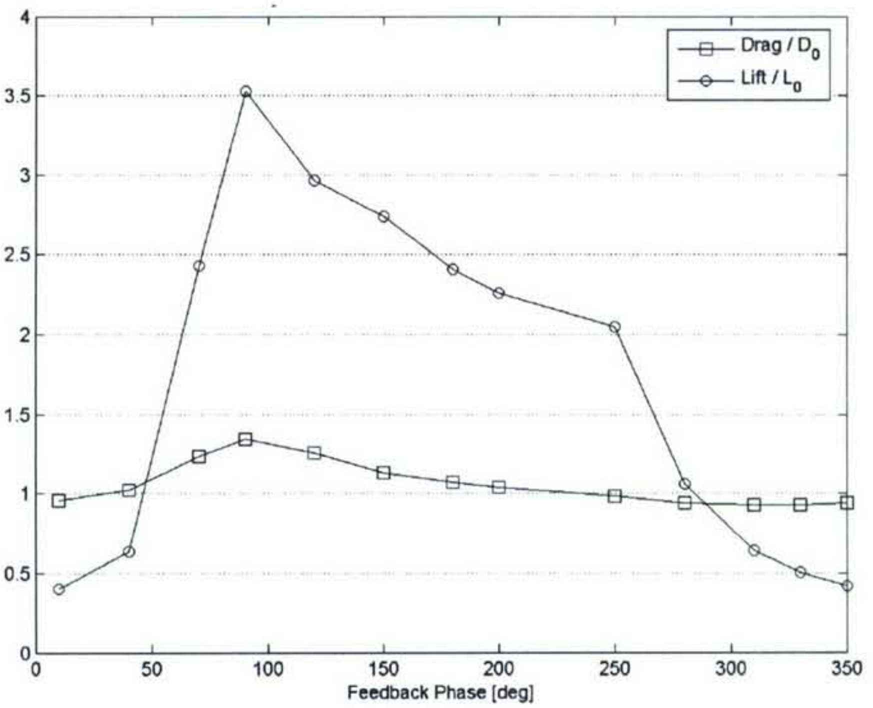


Figure 28: Non-dimensional RMS amplitudes of the stabilized lift and drag forces using single mode feedback (L_0 and D_0 are the RMS amplitudes of the unforced flow's lift and drag force, respectively)

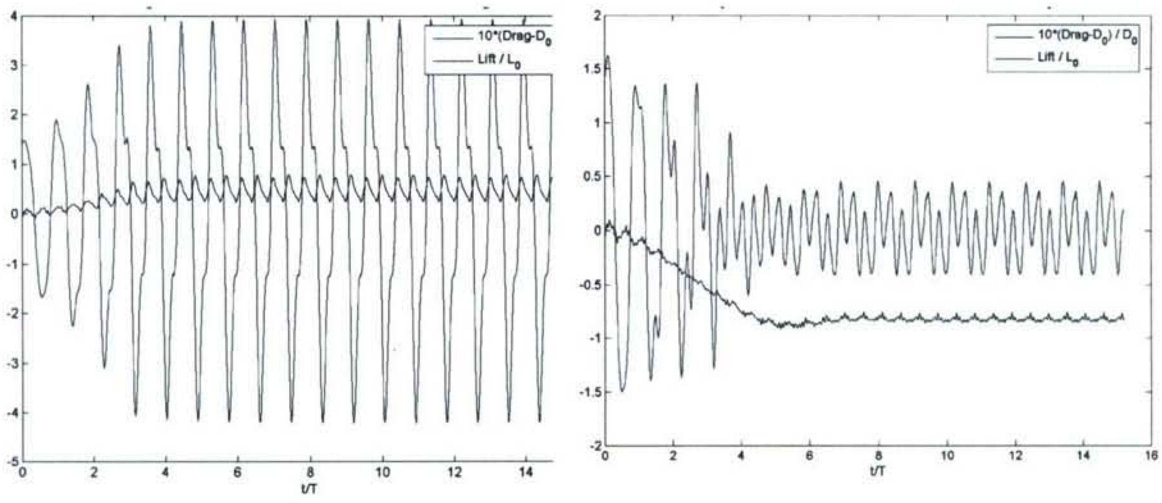


Figure 29: Lift and Drag for feedback using Phase angles 90 degrees (left) and 330 degrees (right), SISO feedback

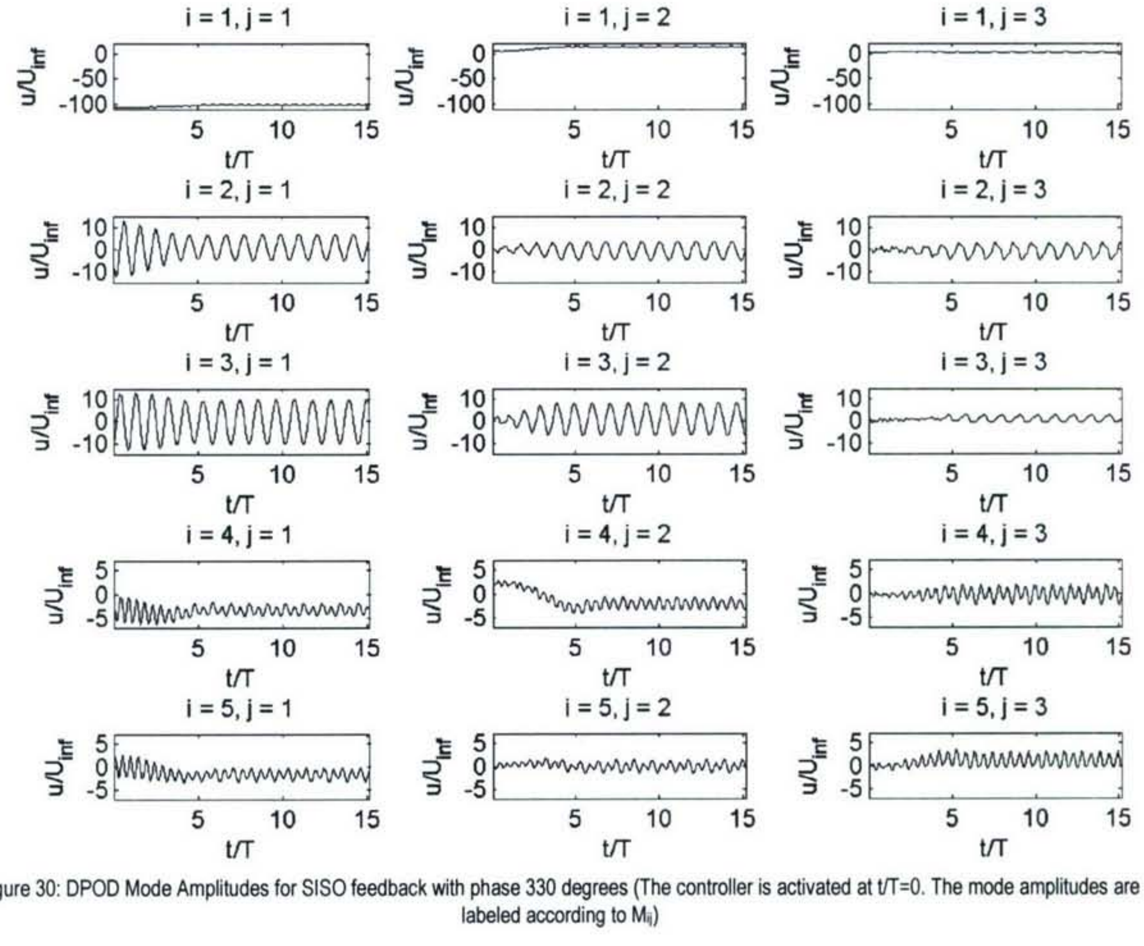


Figure 30: DPOD Mode Amplitudes for SISO feedback with phase 330 degrees (The controller is activated at t/T=0. The mode amplitudes are labeled according to M_{ij})

Figure 31 shows a parameter scan varying the feedback phase of M22, while keeping all other gains as well as the phase of M21 constant. Again a range of detrimental phases exists, where the drag is compared to the new baseline of SISO feedback with 330 degrees phase. However, for a small range of feedback phases between φ=310° and φ=50°, a further reduction in drag beyond SISO control can be observed. This demonstrates the benefit of using MISO control over SISO control. However, the detailed analysis of the MISO control run the led to the largest reduction in drag (Figure 32) shows that there is a lack of stabilization in this type of feedback control. While the drag initially is decreased by more than 6% compared to the SISO level, it is followed by an increase in drag to a stable level about 3% below the baseline. Analyzing the DPOD mode amplitudes for this case (Figure 33), one can see that control of M22 actually destabilizes M21 later in the simulation.

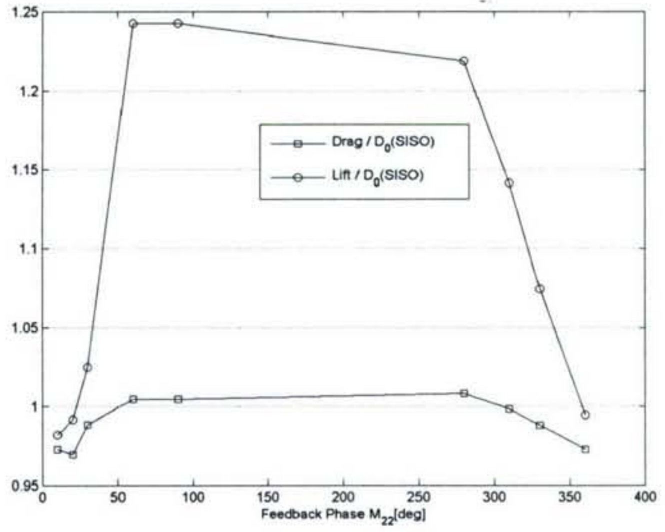


Figure 31: MISO feedback of Modes 21 and 22, with K=0.5e-3 for both modes and a fixed phase of 330 deg for M21. (The feedback phase of M22 is varied, and the resulting lift and drag forces are normalized by those of the SISO feedback of M21 with K=5e-4 and phase = 330.)

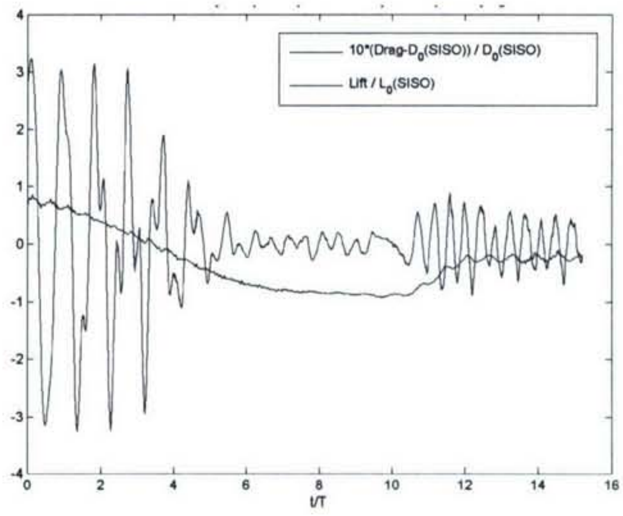


Figure 32: Lift and Drag for MISO feedback with Modes M_{21} and M_{22}, K_{21}= K_{22}=5e-4, φ_{21}=330°, φ_{22}=20 °

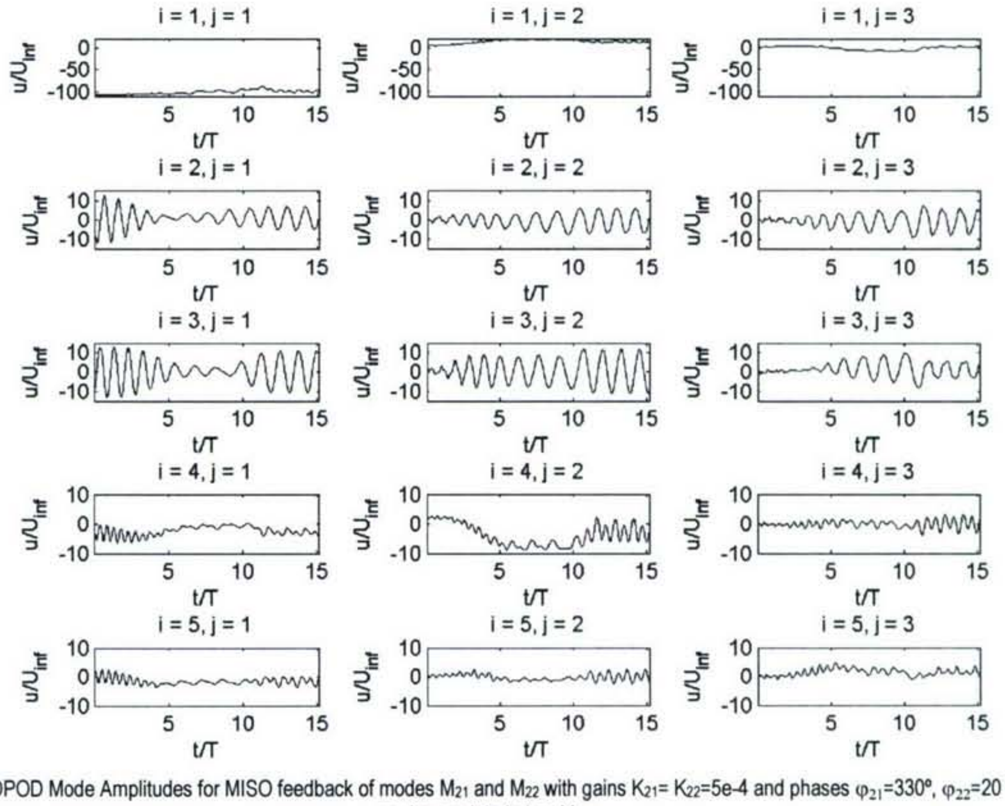


Figure 33: DPOD Mode Amplitudes for MISO feedback of modes M_{21} and M_{22} with gains K_{21}= K_{22}=5e-4 and phases φ_{21}=330°, φ_{22}=20 °. The modes are labeled as M_{ij}

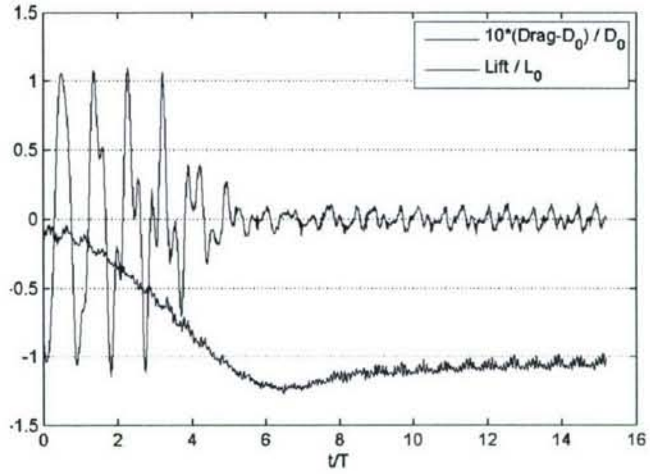


Figure 34: Lift and Drag for MISO feedback with Modes M_{21} and M_{22}, K_{21}= 0.5e-3, K_{22}=0.375e-3, φ_{21}=330°, φ_{22}=20 °

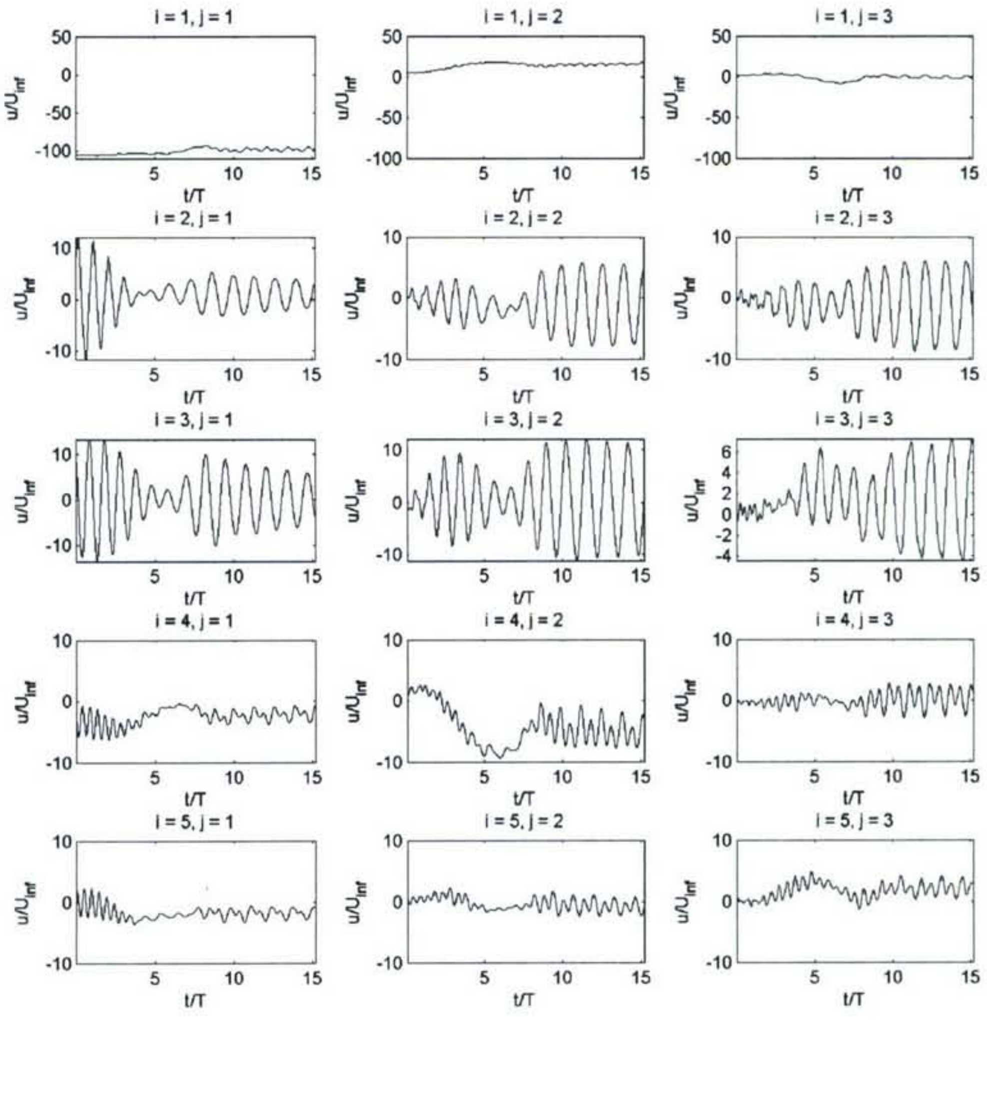


Figure 35: DPOD Mode Amplitudes for MISO feedback of modes M_{21} and M_{22} with gains K_{21}= 0.5e-3, K_{22}=0.375e-3, and phases φ_{21}=330°, φ_{22}=20 °. The modes are labeled as M_{ij}

With some fine tuning of the gains applied to M21 and M22, the flow field can be stabilized using multi mode feedback. Figure 34 and Figure 35 demonstrate this, where the flow is stabilized at a lift fluctuation level more than one order of magnitude smaller than the unforced flow, and at a reduction of drag of more than 10%. However, in this situation the fluctuating amplitude of the shift modes is increased as well – not just for the von Kármán modes, but also for the higher harmonic modes. From a flow physics perspective, this indicates a shift of the vortex formation further downstream as the controller becomes effective in stabilizing the near wake. At the same time, the actuation which remains at the cylinder location becomes less and less efficient in controlling this remaining vortex shedding far away from the cylinder.

4.2 Axisymmetric Bluff Body

To advance our flow control techniques, we applied them to the three dimensional wake behind an axisymmetric bluff body at a Reynolds number of Re=1,500. The investigations started with establishing a lock-in region for the flow field. This is critically important because the structures in the wake only assume the frequency if forcing is applied with frequency/amplitude pairs inside this region. Outside the lock-in region, the wake does not respond to the forcing and meaningful modeling for feedback control is therefore not possible. Once the lock-in region is established, open loop and transient forcing is introduced to study the dynamics of the flow field. The resulting data is then used as the basis of a Low Dimensional Model (LOM) based on Proper Orthogonal Decomposition (POD). With this LOM, flow sensor configurations can be studied to best determine the instantaneous state of the salient features of the flow.

An instantaneous snapshot of the unforced flow field is shown in Figure 36. Near the base of the body, the blowing and suction slots (8 slots distributed evenly around the circumference) are shown in black. Although the axisymmetric bluff body wake shows some similarities to the circular cylinder wake and its von Kármán vortex street, a major difference is that the vortex shedding is not completely periodic, as shown in Figure 37 and Figure 38, where the drag and normal force components, respectively, are plotted as a function of time.

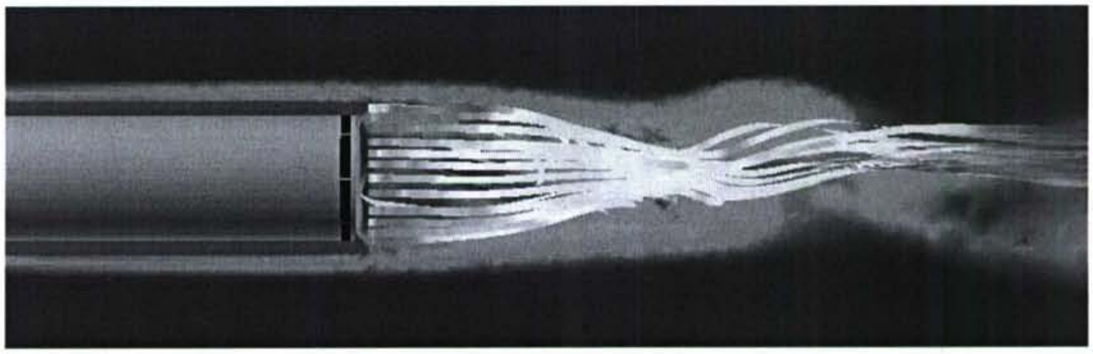


Figure 36: Tail of the axisymmetric body. x-y plane showing instantaneous total vorticity and streamlines colored by total vorticity.

Although the normal force components exhibit long periods of fluctuations at the natural shedding frequency, e.g. between 2s and 5s, the amplitude variations indicate that the shedding is not fully periodic. Another period of similar behavior extends from 8s to approximately 11.5s. It is interesting to note that during these periods of vortex shedding, the drag variations decrease significantly. However, this state is upset by a different mechanism which could be due to the existence of multiple unstable modes in the flow (Schwarz et al. 1994). Another possible explanation, which is not apparent in the force plots, is the slow shift of the symmetry plane of the shed vortices due to slight frequency variations of the mode 1 instability. The power spectrum of the y force component is shown in Figure 39. The peak at f=5.75Hz translates into a Strouhal number St=0.17, matching values reported by Sakamoto (1990). The unsteady wake behavior described above is also evident in the power spectrum plot, which shows a second, smaller peak at f=5.4Hz, and very low frequency content around 1Hz. These slow time variations pose a significant problem when performing POD on the unforced wake. First, determining a time interval in which the structures in the wake are well represented is difficult. Second, and more importantly, because the flow exhibits two apparently different shedding mechanisms, creating POD modes that represent one or the other, which is necessary to distinguish between them, becomes very difficult. And third, although an understanding of the unforced wake is immensely helpful in developing various aspects of feedback flow control, it has to be realized that when feedback control is applied, as is the ultimate goal of this research effort, the unforced state will not be encountered. For these reasons, open loop forcing was applied to study the axisymmetric wake in detail.

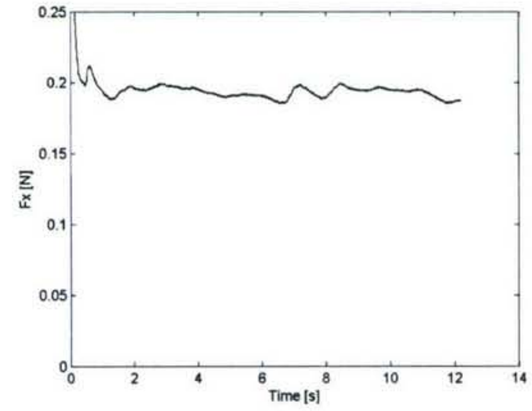
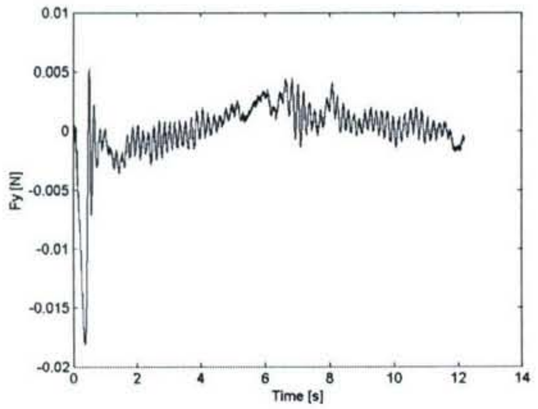


Figure 37: Drag force as a function of time. Simulation started at t=0.



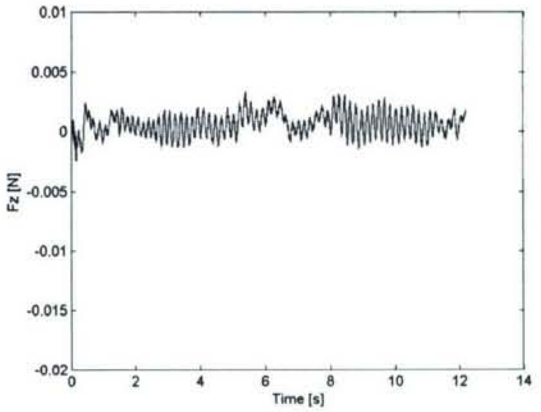


Figure 38: Normal force components Fy and Fz as a function of time. Simulation started at t=0.

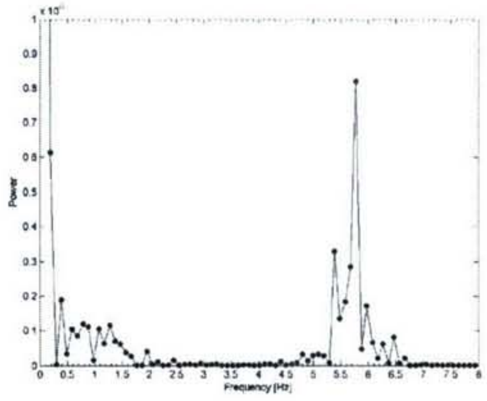


Figure 39: Power spectrum of the unforced wake.

4.2.1 Blowing and Suction Open Loop Lock-In Study

In accordance with the methodology outlined above, a key component of our approach is studying the wake using open loop forcing. The results of open loop simulations will enable the initial development of POD modes representative of the periodic vortex shedding. To introduce forcing, eight blowing and suction slots are located near the base of the body (see Figure 36), which enables forcing up to azimuthal mode k=4. For the present study, azimuthal modes k=±1, which were identified in earlier linear stability analysis as most unstable, are forced at various amplitude levels A_0. The combination of modes 1 and -1 results in a standing wave in azimuthal direction, i.e. the helical character of a single mode k=1 or mode k=-1 wave is masked. By way of an example, results of this forcing applied at an amplitude of $A_0/U_0=0.1$ are shown in the following figures. Figure 40 shows an instantaneous isosurface of total vorticity, Ω. Due to the forcing, the azimuthal phase of the vortex loops is now fixed and the symmetry plane falls in the x-y plane. The symmetry plane is also shown in Figure 41, where the streamwise component of vorticity, ω_x, is plotted at two streamwise planes, x/D=1 and x/D=2.5 downstream of the base. The time history of the three force components on the body are plotted in Figure 42 and Figure 43. Within less than one second or 5 shedding cycles, the drag reaches an almost constant value. As expected, it increases compared to the unforced case (see Figure 37) due to the strengthening of the structures by the applied forcing. The normal forces, when compared to the unforced case, also show a significant change in their behavior. First, as for the drag, the y component of the normal force reaches a periodic state in less than 1 second. And second, the y component is now much larger than the z component because forcing fixes the symmetry plane in the x-y plane (Figure 41).

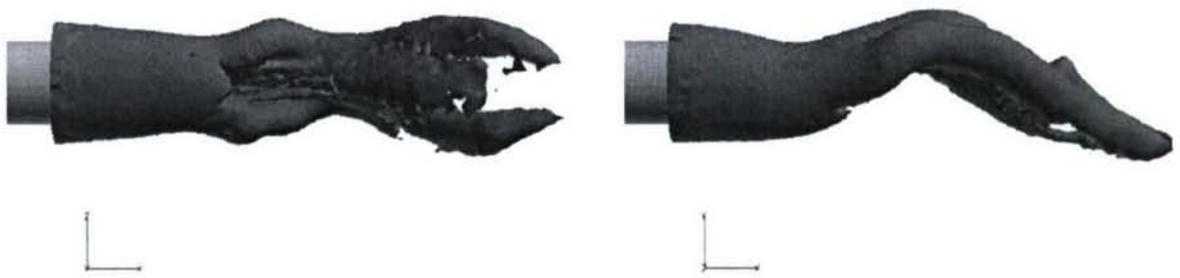


Figure 40: Isosurface of total vorticity Ω=40. Left: view from the positive y-direction. Right: view from the positive z-direction.

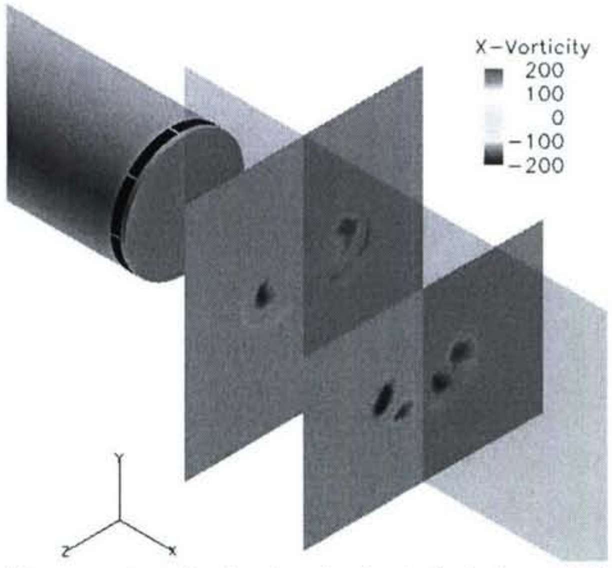


Figure 41: Streamwise vorticity component ω_x at two downstream locations for the forcing case $A_0/U_0=0.1$, $f=f_0$.

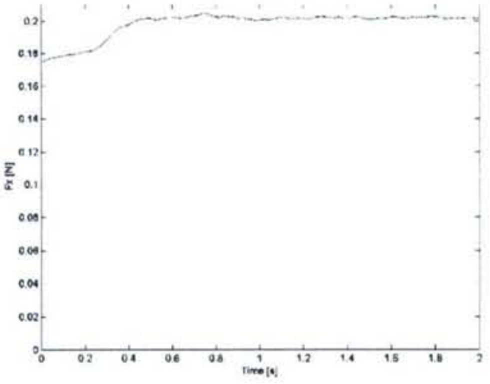


Figure 42: Drag force F_x as a function of time. Forcing started at t=0. Forcing $A_0/U_0=0.1$, $f=f_0$.

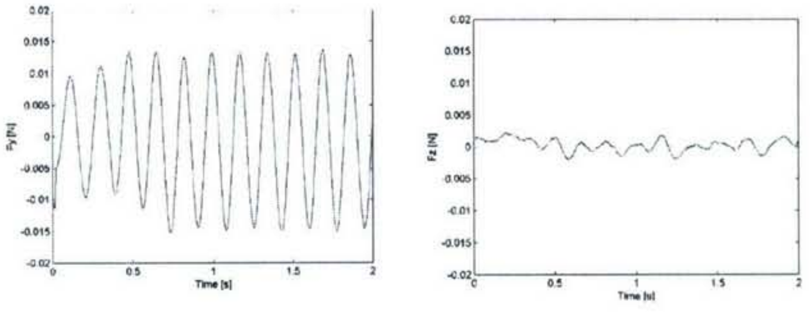


Figure 43: Normal force components F_y and F_z as a function of time. Forcing started at t=0, $A_0/U_0=0.1$, $f=f_0$.

To determine the boundaries of successful lock-in in an amplitude/frequency diagram, the forcing amplitude A_0/U_0 was varied between 2.5% and 30%. In addition, a frequency sweep was performed for normalized frequencies $(f-f_0)/f_0$ from 0 to -0.7 for $A_0/U_0=0.1$ and $A_0/U_0=0.05$, where f_0 is the natural shedding frequency. As a criterion for lock-in, the forces on the body were utilized. If a fixed phase relation between the sinusoidal reference signal and the force components was obtained, lock-in had taken place. The underlying assumption is that the forces on the body are caused by the pressure fluctuations due to the flow structures in the near wake behind the body and are therefore representative of the state of the wake itself. Another way of establishing lock-in is to monitor the velocity fluctuations at a point in the wake (Blevins 1990).

For the case of $f/f_0=1$, $A_0/U_0=0.1$, a comparison between the force components and the streamwise velocity component in the wake is shown below. The drag force is shown in Figure 44. The figure indicates that after about five shedding cycles, the wake locks-in to the forcing, indicated by the periodic time variation of the force. Similar results are observed from the normal force (Figure 45). Comparing the time history of the velocity at (x/D,y/D,z/D)=(8,0.5,0) (Figure 46) with that of the forces shows good agreement in the data in terms of the time it takes for the flow to lock-in to the forcing. While the phase relation is different, lock-in can be observed in all graphs after approximately five shedding cycles.

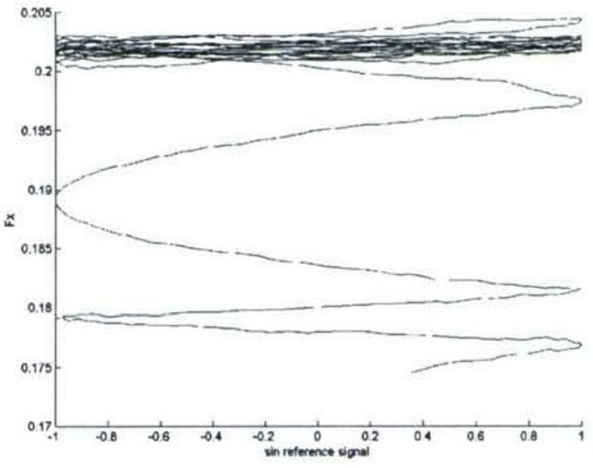


Figure 44: Axial force component as a function of sinusoidal reference signal, f=5.75Hz.

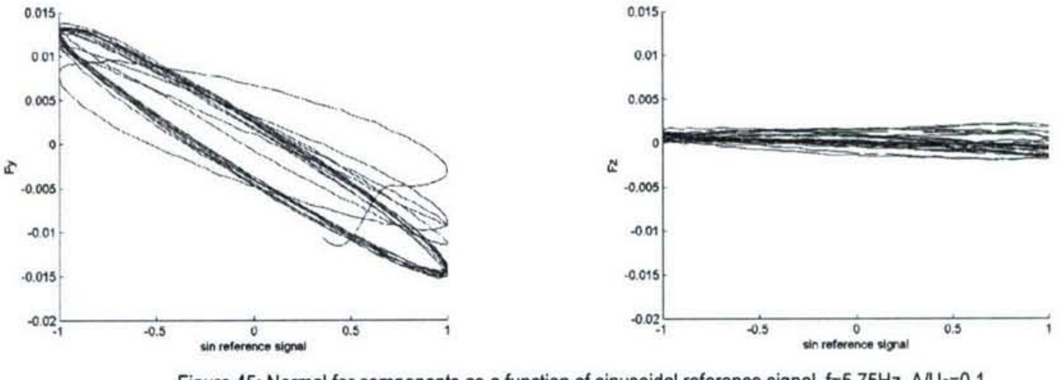


Figure 45: Normal for components as a function of sinusoidal reference signal, f=5.75Hz, $A/U_0=0.1$.

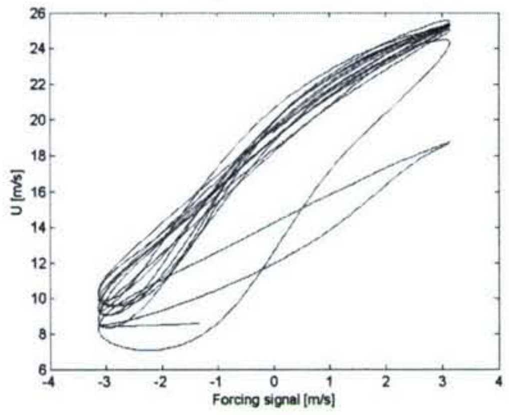
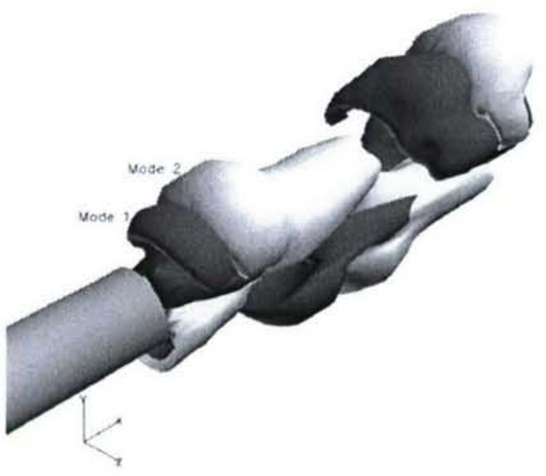


Figure 46: U velocity at (x/D,y/D,z/D)=(1,0.5,0) downstream of the base. Forcing f=5.75Hz, A/U$_0$=0.1.

4.2.2 DPOD model

The complex temporal-spatial behavior of the wake that is evident from Figure 41 suggests the need for powerful low dimensional modeling tools if any feedback control of this flow is to be successful. Developing a meaningful POD based model that is valid not just for one particular orientation of the plane of symmetry of the vortex shedding, like the vertical orientation shown in Figure 41, is difficult. If a single shedding cycle with a given orientation of the plane of symmetry is used to derive POD spatial modes, these modes will feature the same plane of symmetry. An example of this is shown in Figure 47, where one shedding cycle of open loop forced flow data was used to construct POD spatial modes. It can be seen that the spatial modes feature the same plane of symmetry (here the YZ plane) as the flow data. Inspection of the modes reveals that the modes occur in pairs, an effect that is even more evident in the time coefficients shown in Figure 48. This behavior is analogous to the mode pairs observed in POD based models of the circular cylinder wake, where the von Kármán vortex street is represented by two modes that are shifted by 90 degrees, both in time and space. Similarly, Modes 1 and 2, as shown in Figure 47, are of similar shape, but shifted in the downstream direction. The next set of higher order modes, Mode 3 and 4, are of significantly smaller energy content as evidenced by the time coefficient amplitudes, with twice the frequency and half the spatial wavelength when compared to Modes 1 and 2.

While the model used for the derivation of the time coefficients shown in Figure 48 was obtained from a single shedding cycle, data from a transient simulation was projected onto these spatial modes to derive the time coefficients. At time 5.4 s, the orientation of the symmetry plane is perpendicular to the orientation of the shedding cycle used for derivation of the spatial modes. As time goes on, the vortex shedding orientation adjusts to the forcing that was activated at 5.4 s and thus becomes aligned with the spatial modes. Consequently, the amplitudes of the time coefficients are much smaller at 5.4 s, indicating the inability of the model to represent the vortex shedding state that is perpendicular to the modes.



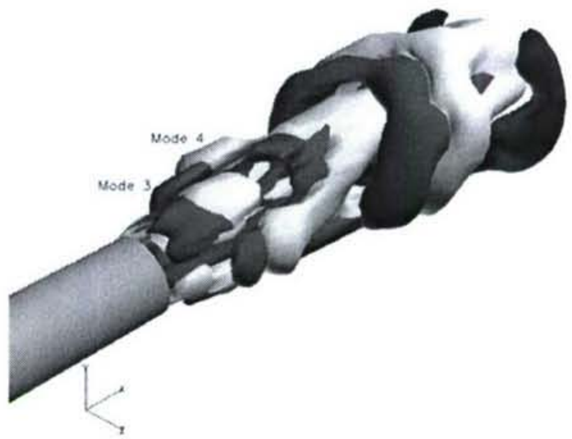


Figure 47: Modes 1 and 2 (left) and 3 and 4 (right), 0.0015, A/U=.1 SPOD from open loop forced simulation.

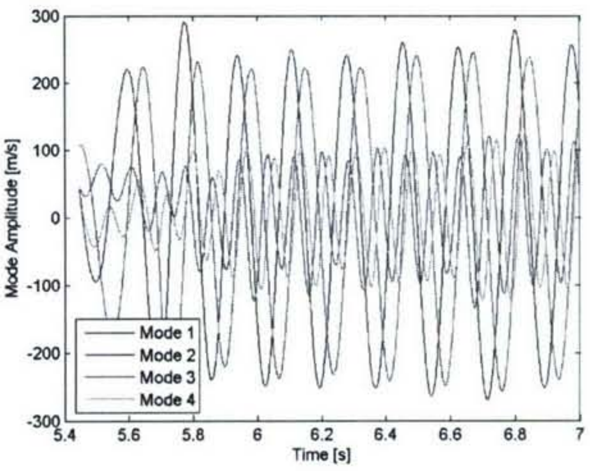


Figure 48: Time coefficients of SPOD[22] model from a single open loop forced shedding cycle, applied to a transient simulation where the spatial orientation of the vortex shedding turns by 90 degrees within the first 3 cycles.

Since the POD modes of a single shedding cycle shown in Figure 47 feature a prominent plane of symmetry, they will only represent vortices well that share the same plane of symmetry. As mentioned above, in the unforced flow, the plane of symmetry shifts very slowly compared to the time scale of the shedding of the vortex loops. As a result, gathering snapshots of vortices with all possible orientations of their symmetry plane becomes a computationally unfeasible task due to the number of simulations required. To bypass this process, we re-sampled the flow field on an axisymmetric grid. The centerline of this grid is aligned with the axis of the body, and the re-sampling grid extends from the base of the model to 6 diameters downstream, with a radius of 3 diameters. 64 azimuthal, 20 radial and 13 streamwise grid points were used. This interpolated set of snapshots of a single open loop forced shedding cycle was then rotated and duplicated four times, forming a snapshot ensemble consisting of one shedding cycle with the original vertical plane of symmetry, and one each with a plane of symmetry rotated by 90, 180 and 270 degrees. The first observation in inspecting the POD spatial modes resulting from this decomposition, which we refer to as four quadrant (4Q) model, is the existence of four (instead of previously two) spatial modes of the fundamental wavelength. Figure 49 shows these spatial modes. The first two of these modes are helical in nature, with a rotation in the clockwise direction when looking downstream. They are of approximately equal energy, as shown in the modal energy in Figure 50. Due to their spatial shift in the downstream direction, they can model traveling vortices in just the same fashion as the two paired modes in the single shedding cycle model presented above. Mode 3 and Mode 4 (not shown) are similar helical modes as Modes 1 and 2, however, they have the opposite rotation direction, i.e. they are turning counter clockwise. If one of the clockwise and one of the counter clockwise rotating modes are plotted superimposed like on the right side of Figure 49, it becomes evident how a clockwise and counter clockwise helical mode together represent the vortex loops that have been reported in literature and which we describe earlier in the paper (see Figure 40).

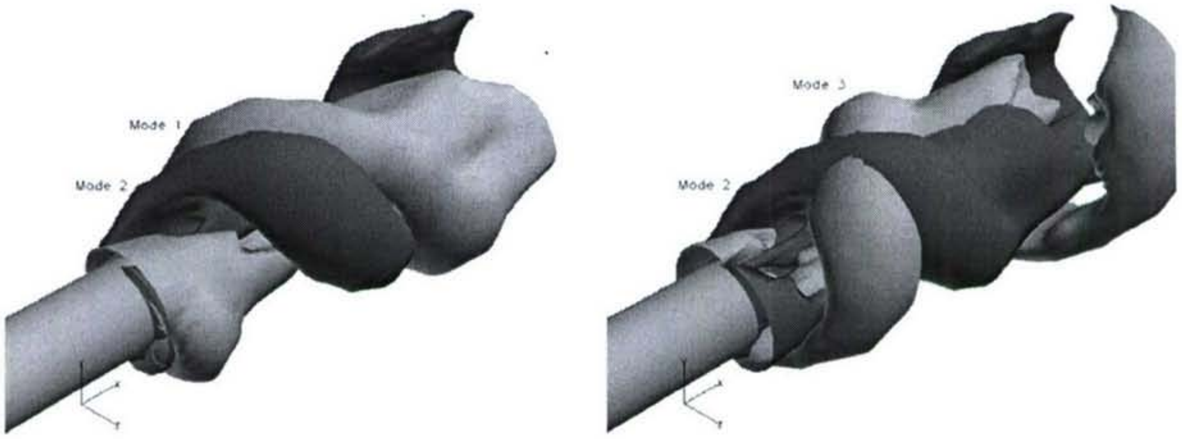


Figure 49: Modes 1 and 2 (left) and 2 and 3 (right), 4Q model.

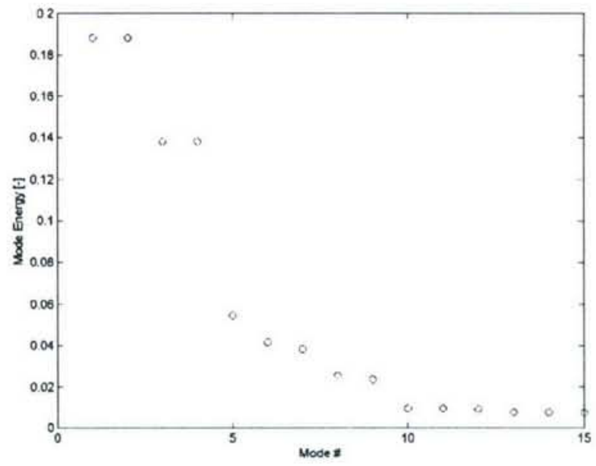
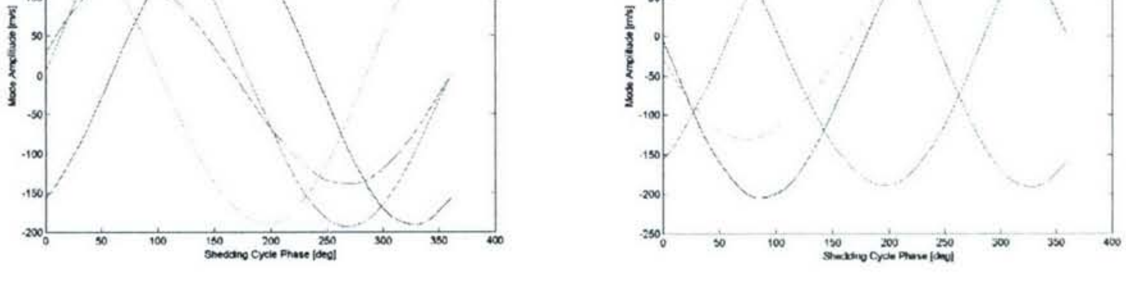
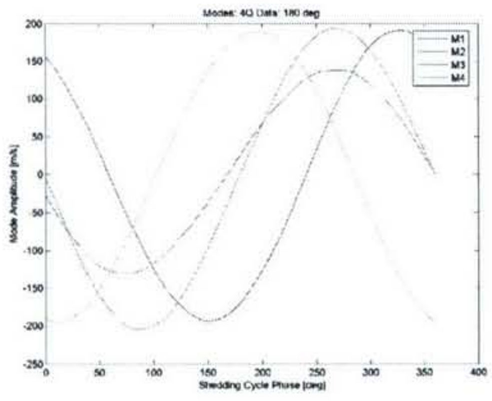


Figure 50: Modal energy, 4Q model.

The traveling wave nature of the mode pairs 1+2 and 3+4 is also evident in the temporal coefficients shown in Figure 51. Modes 1 and 2 always show a phase shift of 90 degrees, no matter what the orientation of the vortex shedding is. The same holds true for modes 3 and 4, which also have a constant 90 degree phase shift for all different vortex shedding orientations. However, the spatial orientation of the vortex shedding can be recovered by analyzing the phase shift of the time coefficients of modes 1 and 4, for example. These modes are in phase for a horizontal plane of symmetry, and out of phase for a vertical plane of symmetry. The orientation of the vortices can also be recovered in a similar fashion by inspection of the time coefficients. Thus, the time coefficients alone are sufficient to determine the phase, frequency and orientation of the vortex shedding.





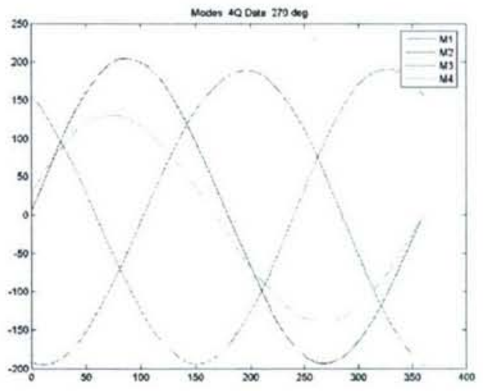
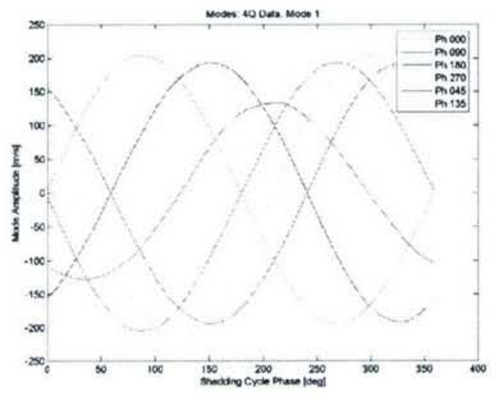


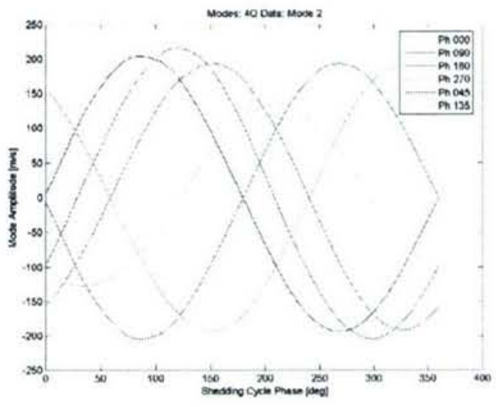
Figure 51 Time Coefficients of first four POD modes of the 4Q model, for shedding orientations of 0, 90, 180 and 270 degrees.

The question remains to be answered if the spatial modes obtained using only four different shedding orientations as done in the derivation of the 4Q model are able to represent shedding orientations that lie in between the original four symmetry plane angles. To investigate this, a look at the temporal coefficients of both on-design cases (symmetry plane orientations of 0, 90, 180 and 270 degrees) and worst possible off-design cases (shedding orientations of 45 and 135 degrees) is helpful. Figure 52 demonstrates that the time coefficients of the 45 degree shedding orientation fall right between those of 0 and 90 degree orientation, and similarly the 135 degree shedding orientation lies between 90 and 180 degrees in phase. In addition, the amplitudes of these time coefficients are comparable to those of the shedding orientations that were used for model derivation, indicating that the energy of the off-design shedding cases is captured to the same degree as that of the design cases. This indicates that by using just four individual shedding orientations for model derivation, all possible orientations through the full 360 degrees are captured with equal accuracy. For completeness of discussion, a set of modes derived from only two shedding cycles oriented in the 0 and 90 degree directions failed to achieve this range of validity.

a.) Mode 1



b.) Mode 2



c.) Mode 3

d.) Mode 4

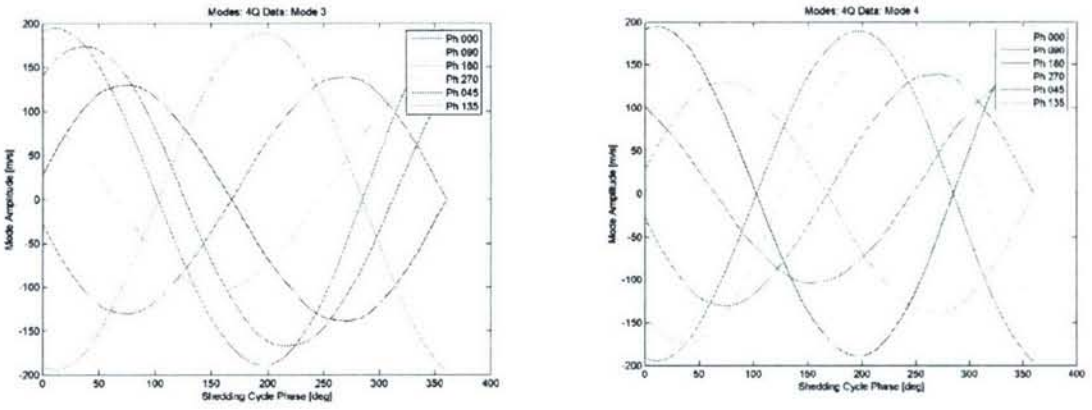


Figure 52: Time Coefficients for 4Q model of vortex shedding with 45 degrees and 135 degrees orientation.

4.3 Turbulent Circular Cylinder Wake

Extending the flow control methods described above to higher Reynolds numbers and turbulent flows, we investigated the cylinder wake at a Reynolds number of Re=20,000 computationally as well as experimentally. An instantaneous snapshot of the wake is shown in Figure 53.The simulations were performed using a Mach number of M=0.1. The length to diameter ratio of the cylinder was L/D=4. The diameter of the cylinder was D=2m. Periodic boundary conditions were used on the computational surfaces at the cylinder ends, modified Riemann invariants were used as a farfield boundary condition, whereas a no slip, adiabatic wall was employed for the cylinder surface. A time step of $\Delta t=0.576 \cdot 10^{-3}$ seconds corresponding to $\Delta tU/D=0.02$ non-dimensional time steps was used. Time was non-dimensionalized by D/U where the D is the cylinder diameter and U is the free stream velocity. The total simulation time was 10,000 time steps, corresponding to 20 cycles of vortex shedding. The temporal resolution of the simulations was 500 time steps per von Karman shedding cycle. In order to increase the stability, an advection damping coefficient of 0.01 was used in the computations. No damping was used for diffusion. As an initial perturbation to trigger the unsteadiness in the flow simulations, the incoming flow was skewed by an angle of attack of 1 degree.

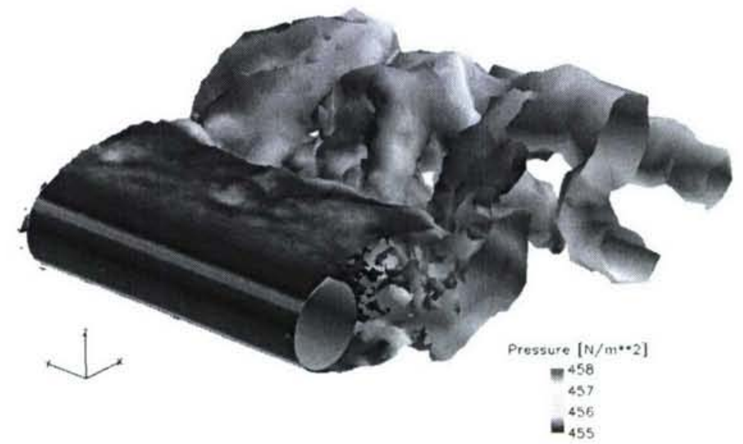


Figure 53: Instantaneous isosurface of total vorticity, ω=20 1/s, colored by pressure.

The grid is an unstructured grid consisting of clustering of prismatic cells in the boundary layer and tetrahedral cells outside the boundary layer. The minimum cell height is $1x10^{-3}$ m, which corresponds to $5x10^{-4}$ when non-dimensionalized by the cylinder diameter. $y^{+}_{average}$ (average non-dimensional first cell height for the boundaries) is around 0.3. The grid has 879,603 cells. The farfield boundaries are 20 diameters away from the cylinder surface. With a computational time step of $\Delta t=1.5\cdot10^{-3}$ seconds and a shedding frequency of about f=7.2 Hz, the Strouhal number of the von Kàrmàn vortex shedding for the present data is about St=fD/U=0.21. The temporal resolution of the simulation was about 380 time steps per von Karman shedding cycle. For the purpose of this study, we obtained a time series of 1,200 snapshots using every fifth computational time step from the converged (periodic) solution, thus arriving at a data set containing about 13 shedding cycles of the von Karman mode.

The coordinate system used in the computations and the grid around the cylinder surface are shown in Figure 54a; x is the streamwise direction and the y axis is aligned with the cylinder axis. The origin is at the near end of the cylinder in the cylinder center. Figure 54 shows the surface grid, a plane normal to the cylinder axis, and a zoom-in near the surface.

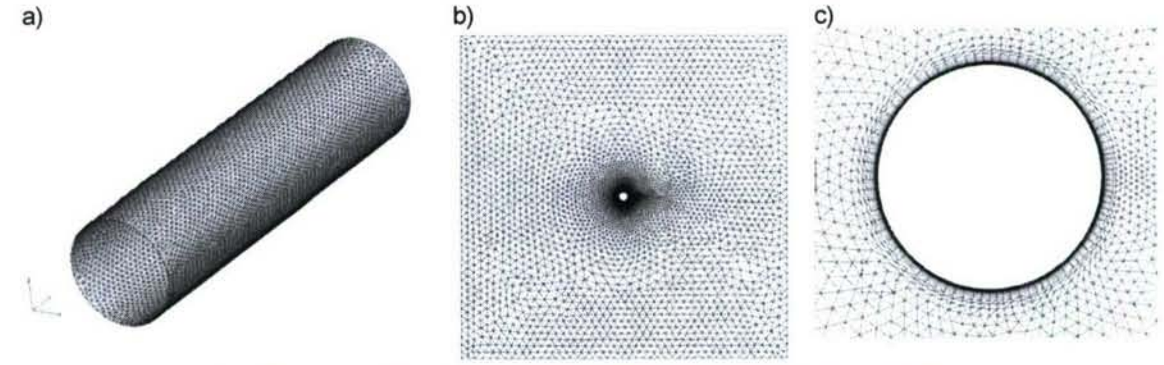


Figure 54: a) Surface mesh, b) mesh at the periodic boundary plane, c) zoom-in near the surface.

4.3.1 DPOD based low order modeling

For low-dimensional control schemes to be implemented, a real-time *estimation* of the DPOD modes present in the wake is necessary, since it is not realistic to measure the modes directly and be able to close the loop. Velocity field data, provided by the CFD simulation, is fed into the DPOD procedure as described in the previous section. Then, the estimation of the low-dimensional DPOD mode amplitudes is provided using an appropriate estimator. Sensor measurements may take the form of wake measurements such as velocity, or, as in this effort, of body mounted pressure measurements. This process leads to the identification of the measurement equation, required for design of the control system. For practical applications it is desirable to reduce the sensors required for estimation to the minimum.

The requirement for the estimation scheme is to behave as a modal filter that "combs out" the higher modes. The main aim of this approach is to thereby circumvent the destabilizing effects of observation "spillover" as described by Balas (1978). Spillover has been the cause for instability in the control of flexible structures and modal filtering was found to be an effective remedy (Meirovitch 1990). The intention of the proposed strategy is that the signals, provided by a certain configuration of sensors placed in the wake, are processed by the estimator to provide the estimates of the first six DPOD mode amplitudes across 9 *x-z* planes (*y*=0,0.5,1,…,4). An estimation scheme, often used in flow control studies is the linear stochastic estimation (LSE) procedure introduced by Adrian (1977). The LSE of low dimensional mode amplitudes was successfully applied to the unforced Ginzburg-Landau wake model (Cohen et al. 2004) and the unforced circular cylinder wake at low Reynolds numbers (Cohen et al. 2006). A major challenge lies in finding an appropriate number of sensors and determining locations that best enable the desired modal filtering.

The intent of the proposed strategy is that the surface pressure measurements provided by the body mounted pressure sensors are processed by the estimator to provide the estimates of the first six DPOD mode amplitudes. All the measurements were taken after ensuring that the simulation of the cylinder wake flow regime converges to physically reasonable unforced behavior. The DPOD mode amplitudes, $\alpha_{i,j}$(i=1,2,3 & j=1,2) will be mapped onto the extracted sensor signals from the pressure sensors, P_s, as follows:

$$\hat{\alpha}_{i,j}(t) = f(P_s(t), \hat{\alpha}^*_{i,j}, P^*_s) \qquad (1)$$

where $\hat{\alpha}_{i,j}(t)$ is the estimate of the 6 DPOD modes for i=1,2,3 and j=1,2; $P_S(t)$ is the filtered sensor signal from 's' number of surface pressure sensors; $\hat{\alpha}^*_{i,j}$ and P^*_s are the regression matrices of past DPOD estimates and filtered pressure signals respectively; *f* is the non-linear mapping between the present and past sensor readings and past DPOD estimates.

The issue of sensor placement and number has been dealt with in an ad-hoc manner in published studies concerning closed-loop flow control (Cohen et al. 2006). For effective closed-loop control system, the following questions need to be answered:

- How many sensors are required?
- Where will the sensors be placed?
- What are the criteria for judging an effective sensor configuration?
- What are the robustness characteristics of a given sensor configuration?

In this effort, an attempt will be made to emulate some of the proven successes from the field of structural control.

Heuristically speaking, when some very fine dust particles are placed on a flexible plate, excited at one of its natural frequencies, after a short while the particles arrange themselves in a certain pattern typical of those frequencies. The particles will be concentrated in the areas that do not experience any motion (the nodes). On the other hand, the areas where the motion is large (the internodes) will be clean of particles. It is at the internodes that the vibrational energy of a particular mode is at a maximum and sensors placed at these locations are extremely effective in estimating that particular mode (Bayon de Noyer 1999).

The above heuristic approach has been used by Bayon de Noyer (1999) in finding effective sensor placements for acceleration feedback control to alleviate tail buffeting of a high performance twin tail aircraft. Note the usage of the term "effective sensor placements" as it is based on validated heuristics as opposed to "optimal sensor configuration" that results from a mathematically optimal pattern search for a sensor configuration. What needs to be done to determine an effective sensor configuration is to find the areas of the most energetic modal activity.

CFD data provided pressure signals at 151 x 201 locations distributed in a structured fashion along the entire surface of the simulated cylinder (see the cylinder surface grid in Figure 54a). A three-step procedure is proposed for determining sensor placement and number as follows:

- Run a simple POD procedure, after removing the mean pressure mode, on the 151 x 201 pressure signals. The first two surface pressure POD modes contain 85% of the total "energy". As much as 58% of this is in the first periodic mode, whereas the second seemingly non-periodic pressure mode has approximately 27% of the energy. Furthermore, the frequency of the first mode correlates to the Strouhal shedding frequency as seen in Figure 55 which depicts the mode amplitudes of the first two pressure modes. The spatial eigen functions of the first two modes are illustrated in Figure 56.
- The spatial Eigenfunctions obtained from the above POD procedure provides information concerning the locations where the modal activity is at its highest (see Figure 56). Examine the maxima/minima of the spatial Eigenfunctions of the surface pressure grid.
- Place sensors as detailed in Table 4 which correlates to the energetic maxima and minima of the first two POD modes as illustrated in Figure 55. The locations of the sensors in Table 4 are referenced in terms of both grid definition as well as global coordinates, non-dimensionalized with the diameter, D. θ, the circumference angle, has its origin in the mid-cylinder plane at the trailing edge and θ advances along the circumference in an anti-clockwise fashion. The first two sensors target the DPOD periodic modes associated with the Kármán shedding frequency, whereas, the two other sensors target the nonperiodic DPOD modes. The filtered time histories of the four pressure sensors are provided in Figure 57.
- Finally, an appropriate non-linear estimator needs to be developed that will map real-time pressure signals from the above four surface sensors to the 6 DPOD wake velocity modes. Initial work reported by Cohen et al. (2006) on a neural network estimator (Nørgaard et al. 2000, Cohen et al. 2006), was based upon unforced experimental data.

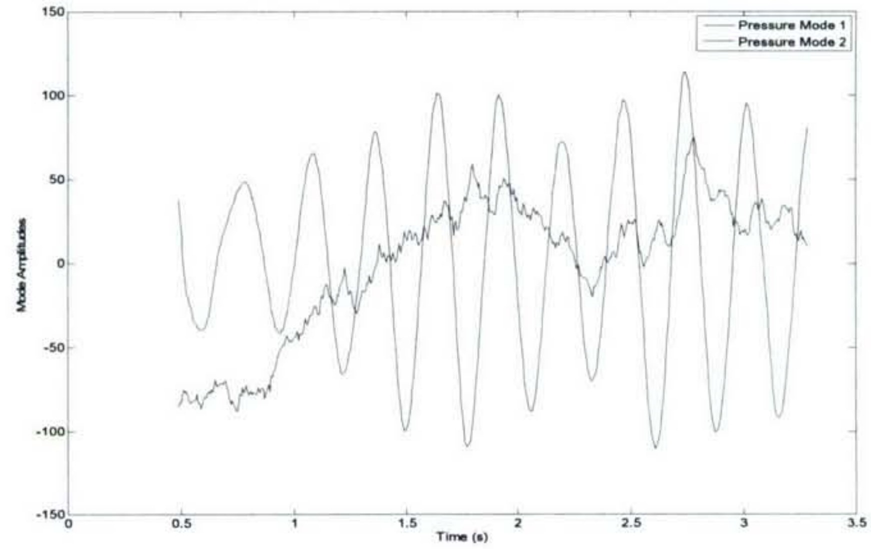
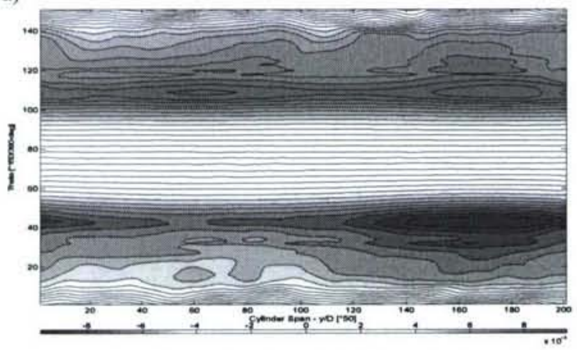


Figure 55: Surface pressure POD mode amplitudes of the first two modes.

a)


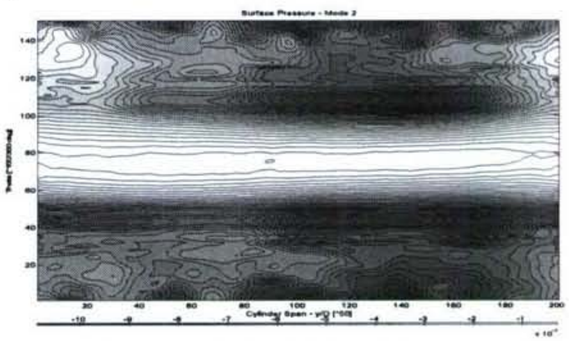
b)


Figure 56: Spatial Eigenfunctions of the surface pressure POD Modes, (a) von Kármán Mode 1; (b) Mode 2.

Sensor #	Sensor Coordinates	
	Span (y/D)	θ (deg)
1	6.8	264
2	6.8	108
3	5.6	0
4	0.6	7

Table 4: Sensor locations.

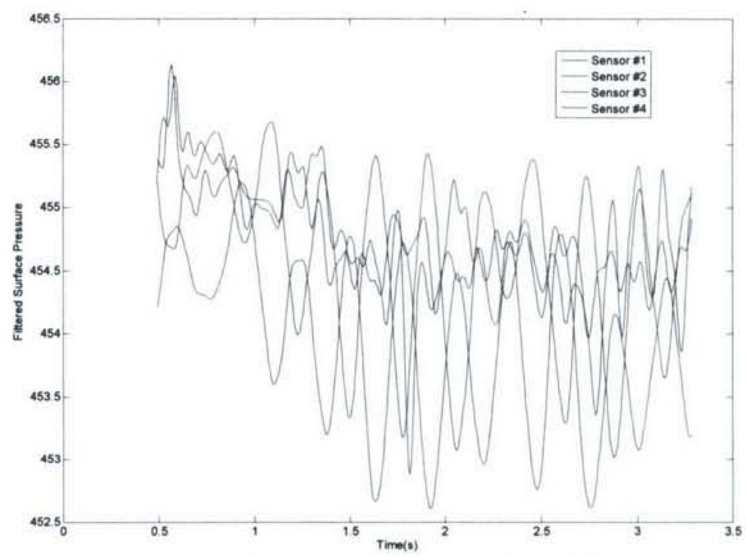


Figure 57: Filtered time histories of the sensor configuration described in Table 1.

The surface pressure data as well as the wake streamwise velocity data were analyzed using Proper Orthogonal Decomposition (POD). For the wake, the data was further scrutinized using Double POD (DPOD).

In the current investigation, these bins are defined by the maxima in the section lift force as shown in Figure 58 for y/D=2. The data shown in the figure indicate that a discernable frequency of the vortex shedding is established around t=0.5s. The other spanwise planes show similar, but not identical behaviour. It is exactly the slight variation between the spanwise locations that is of interest in the current investigation of the three-dimensional development of the cylinder wake.

The spatial distribution of the DPOD modes obtained for an analysis with I=5, J=3, i.e. five main modes (mean flow, the two von Kármán modes, and the higher mode pair) and their respective first and second shift modes, is plotted in Figure 59. Modes (2,1) and (3,1) are the spatial components of a travelling vortex street, indicated by the streamwise shift of their respective maxima by half a wavelength. Figure 60 shows that the mean flow mode (1,1) contains the majority of the energy, followed by the unsteady von Kármán modes, (2,1) and (3,1). The spatial modes as well as the energy distribution is very similar to the results found for a laminar cylinder wake at *Re*=100.

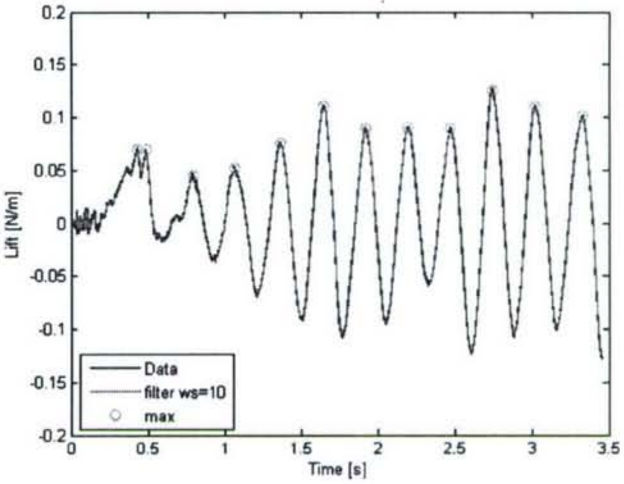


Figure 58: Section lift force at center plane y/D=2.

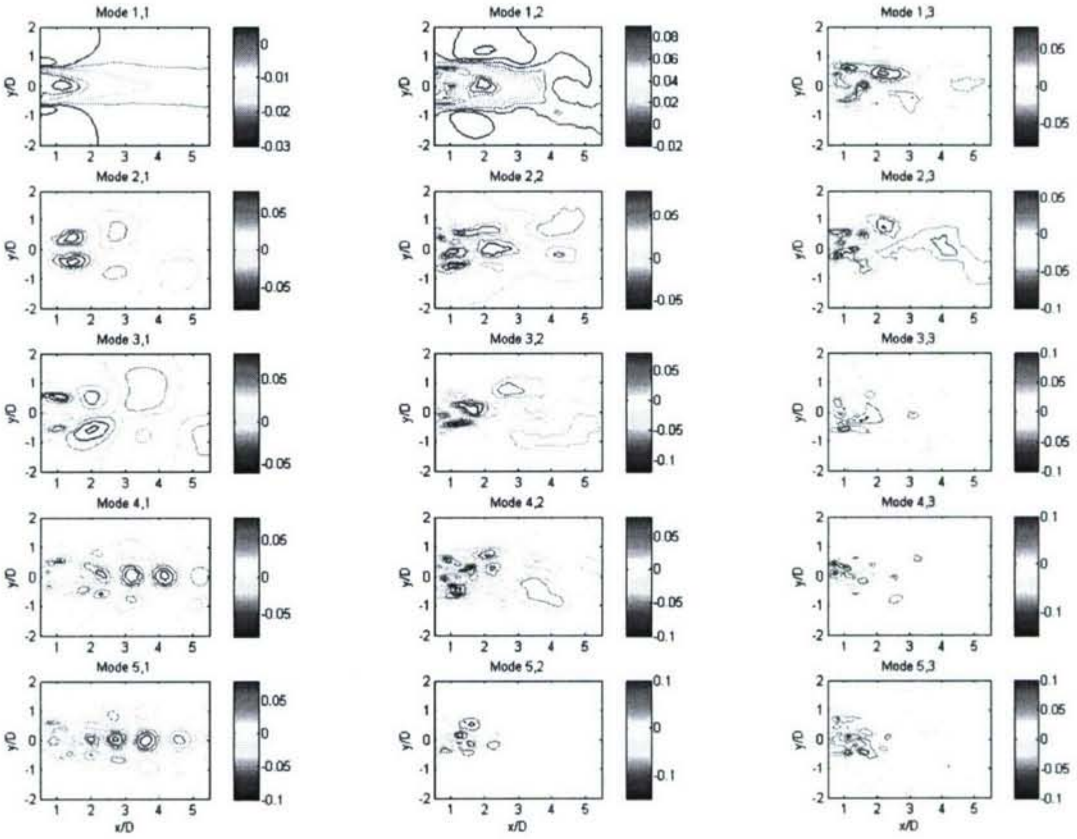


Figure 59: Spatial DPOD modes of the 5x3 model. Mode (1,1) is the mean flow, modes (2,1) and (3,1) represent the von Kármán vortex street.

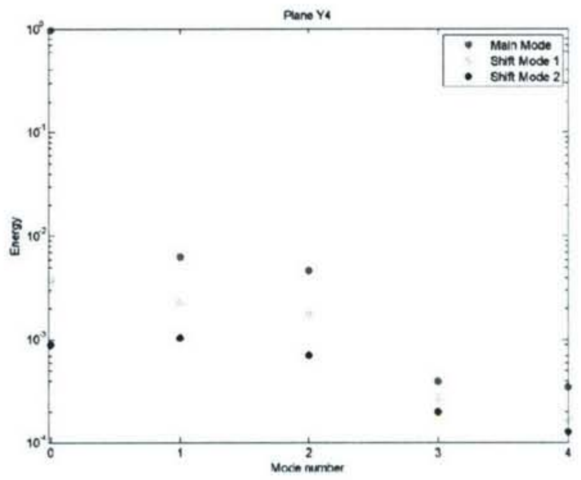


Figure 60: DPOD mode energy.

The mode amplitudes for the von Kármán modes and their first shift modes are given in Figure 61. In conjunction with the spatial modes shown in Figure 59, the mode amplitudes, which are phase shifted by 90°, confirm that modes (2,1) and (3,1) represent a vortex street. In addition, the variation of the maximum amplitude of the oscillations as well as the variations in the mode amplitudes for modes (2,2) and (3,2) over time indicate an energy transfer that has not been observed in the low Reynolds number wake. In contrast to the *Re*=100 case, the shift modes do not converge to a relatively steady value, confirming that the wake undergoes constant changes that seem to prevent fully developed limit cycle oscillations and therefore a fully periodic flow.

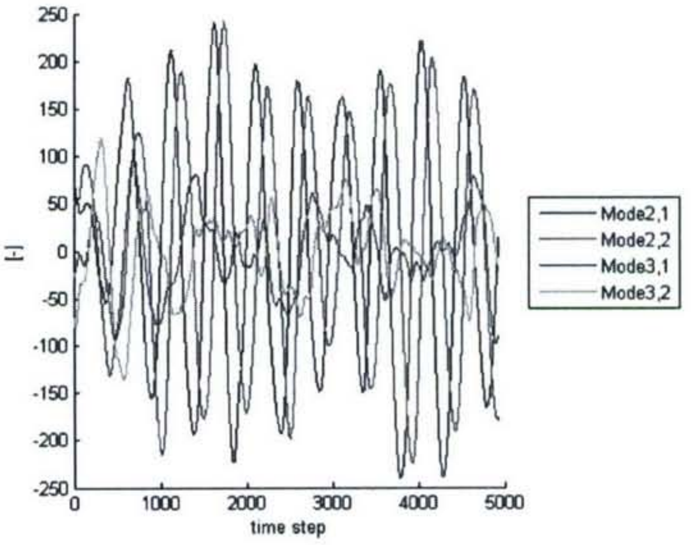


Figure 61: Mode amplitudes of von Kármán modes (2,1) and (3,1) and their first shift modes.

4.3.2 Neural Network based DPOD Mode Amplitude Estimation

In this effort, our estimation method of choice is based on a nonlinear system identification approach described by Nørgaard et al. (2000) using Artificial Neural Networks (ANN) and ARX models. For the mapping of velocity measurements provided by the sensors onto the mode amplitudes of the six DPOD modes, the modified NNARXM (Neural Network Autoregressive, eXternal input, Multi output) algorithm, originally developed by Nørgaard et al. (1990) is used as ANNE (Artificial Neural Network Estimation).

The decision was to look into universal approximators, such as artificial neural networks (ANN), for their inherent robustness and capability to approximate any non-linear function to any arbitrary degree of accuracy. The ANN, employed in this effort, in conjunction with the ARX model is the mechanism with which the dynamic model is developed using the POD mode amplitudes extracted from the CFD simulation. Non-linear optimization techniques, based on the back propagation method, are used to minimize the difference between the extracted POD mode amplitudes and the ANN while adjusting the weights of the model. In order to assure model stability, the ARX dynamic model structure is incorporated. This structure is widely used in the system identification community. A salient feature of the ARX predictor is that it is inherently stable even if the dynamic system to be modeled is unstable. This characteristic of ARX models often lends itself to successful modeling of unstable processes.

ANNE, based on the Multilayer Perceptron Neural Network, uses an adequate training set which comprises of 300 time steps (approximately 5.5 shedding cycles). The ANNE procedure is developed separately for each for the 9 streamwise planes, which are equidistant at 0.5D and stretching 4D (the entire computational span). The ANN network is designed using the identical sensor time histories and then the static ANN design with fixed weights, with its associated weighing matrices, is validated with validation data that was not used for training comprising of 186 time steps (approximately 3.5 shedding cycles). The purpose is to obtain a robust and real-time estimator for as low a number of sensors as possible for application to wake control.

The artificial neural network (ANN) has the following features:

- **Input Layer:** Comprised of two types of data. One coming from the surface mounted pressure sensors and the other from past DPOD estimates. Each of the four pressure sensors each being transformed into 8 inputs (t, t-1, t-2.... t-7). Then, each of these inputs has 6 time delays (i.e ~ ¼ of a shedding cycle retained in the regressor matrix comprising of past sensor information). This approach to manipulation of sensor input was found to be effective by Nørgaard et al. (1990). Additionally, 8 past DPOD estimates (i.e. ~ 1/7 of a shedding cycle) for all 6 DPOD modes are utilized. Therefore, the number of neurons in the input layer is:

$$\underbrace{4}_{\#\text{ of Sensors}} \times \overbrace{8 \times 6}^{\text{Sensor Time Delays}} + \underbrace{6}_{\#\text{ of DPOD Modes}} \times \overset{\text{DPOD Estimates Time Delays}}{8} + \underbrace{1}_{\text{Bias}} = 241$$

- **Hidden Layer:** One hidden layer consisting of 12 neurons. The activation function in the hidden layer is based on the non-linear *tanh* function. A single bias input has been added to the output from the hidden layer.
- **Output Layer**: Six outputs, namely, the 6 DPOD mode amplitudes: the mean flow, $\alpha_{1,1}$, the two fundamental von Kármán POD periodic modes, $\alpha_{2,1}$ and $\alpha_{3,1}$ and the three associated shift modes, namely, $\alpha_{2,1}$, $\alpha_{2,2}$ and $\alpha_{3,2}$. The output layer has a linear activation function.
- **Weighting Matrices**: The weighting matrices between the input layer and the hidden layer (W_1) and between the hidden layer and the output layer (W_2) depend on the number of sensors. For each of the 9 planes, a common ANN architecture is maintained, whereby, W_1 is of the order of [241×12] and W_2 is of the order of [13×6]. These weighting matrices are

initialized randomly and the weighing matrices for each of the nine planes are unique following the uniqueness in the DPOD modes per plane.

- **Training the ANN:** Back propagation, based on the Levenberg-Marquardt algorithm, was used to train the ANN using the toolbox by Nørgaard et al. (1990). The training procedure converged in approximately 30 iterations. The training set contained 300 snapshots.
- **Validating the ANN:** The validation set contained 186 snapshots.

The resulting DPOD estimates for each of the nine planes look very good. Figure 62 and Figure 63 present the ANNE estimates for the DPOD mode amplitudes for planes y/D=0.5 and y/D=3.5. The validation phase starts at 2.12 seconds. Although there are some errors for the estimates in the validation phase as compared with the training phase, these errors are very small. The ANNE estimates for all 9 planes show errors of a similar order, although the estimates of just two planes have been included in this paper for space considerations. Note that the periodic modes have been captured exceedingly well with respect to frequency, phase and amplitude.

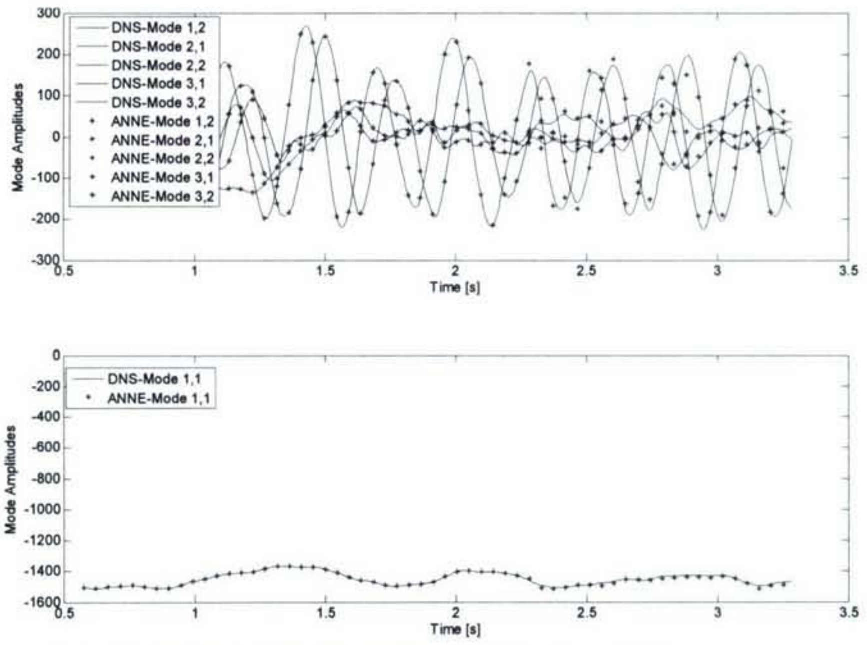


Figure 62: DPOD Mode Amplitude Estimates using ANNE, plane y/D=0.5.

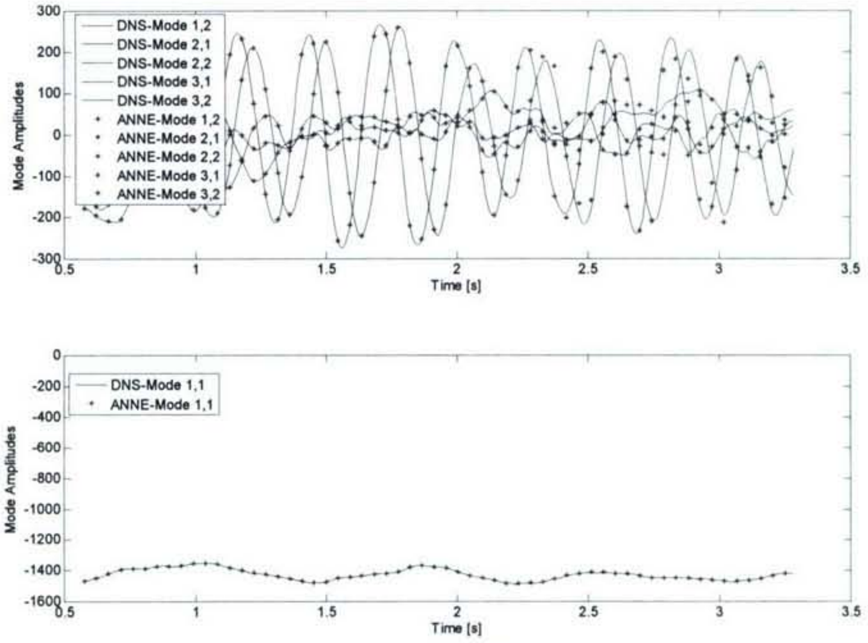


Figure 63: DPOD Mode Amplitude Estimates using ANNE, plane y/D=3.5.

4.4 Body Forcing of a Circular Cylinder wake

The capability to apply body forcing was developed during this program to simulate the effect of plasma actuators on a given flow field. As the other feedback flow control methods, the actuation is completely controllable from within the developed feedback flow control framework, i.e. using the Cobalt/Matlab interface. During the simulations, the location, shape, as well as the force strength and direction can be altered at every time step. To test this new functionality, the cylinder wake at Re=20,000, which was introduced above, was used. Figure 64 shows a comparison of the instantaneous wake structure for both the unforced and forced cases. Unsteady open-loop volume forcing is applied in phase at the top and bottom of the cylinder (Figure 65), using a forcing frequency of f=35Hz, approximately ten times the natural shedding frequency. Although a comprehensive comparison has not been performed to date, the figure clearly shows the effectiveness of the forcing and the feasibility of using a volume force as a model for plasma actuators. Research is ongoing in developing a quantitative model of plasma actuators.

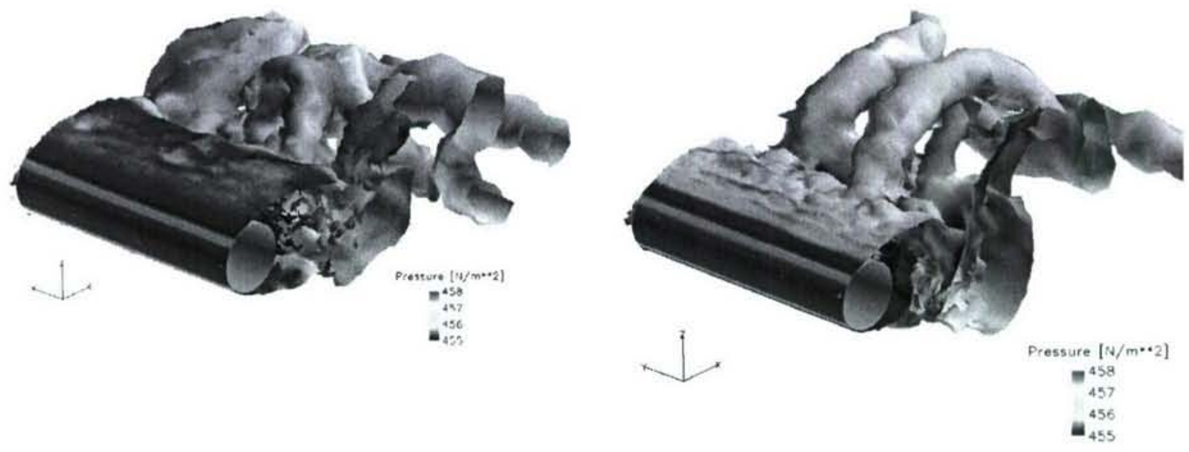


Figure 64: Comparison of the unforced (left) and forced (right) cylinder wake. Instantaneous isosurface of total vorticity, ω=20 1/s, colored by pressure.

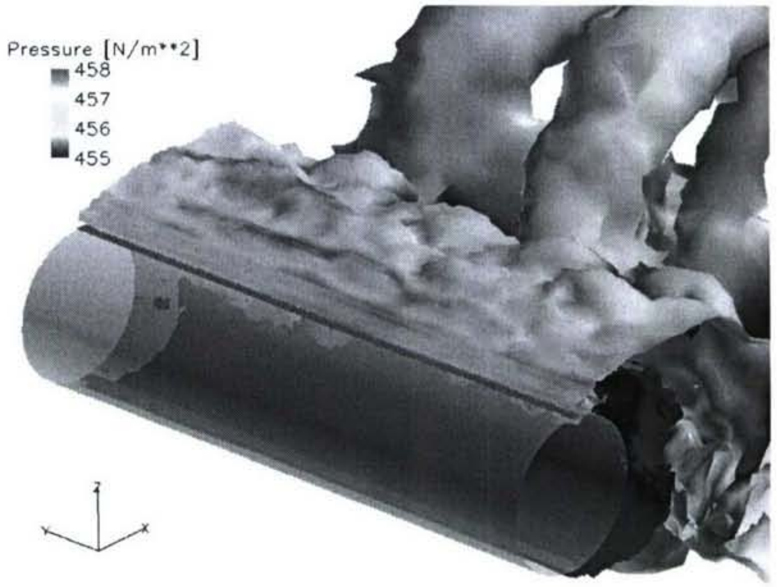


Figure 65: Plasma forcing strips (red) for cylinder wake control.

5 Summary

This program succeeded in achieving the goal set in the original proposal solicitation, which is the development of a software toolbox to enable feedback flow controller development. Our toolbox is comprehensive in that it covers all aspects of controller development. Starting with data collection of the unforced and open loop forced flow response, through low dimensional model development, controller design and testing all the way to controller verification against a truth model, all important aspects of controller development are covered. Since all feedback control related software is implemented in Matlab, the controller developer can easily implement any control algorithm conceivable, using one of the most popular state of the art 4th generation programming languages. At the same time, computationally intensive algorithms were parallelized in order to improve execution speed and allow for processing of even the largest data sets. This was done using the highly efficient open source file format hdf5, in order to allow low level access to the data by the user if needed. Based on feedback from industry representatives at the AFOSR contractors meeting in Long Beach earlier this month, we expect the toolbox to become an indispensible tool for controller development not just for academia, but also for applied problems encountered by the industry. The toolbox developed around the Cobalt flow solver allows for physically correct modeling of any flow control mechanism available today, by using rigid body motion, blowing and suction boundary conditions and body forces. This can be done both open loop, closed loop and even combinations of multiple controllers and different actuation mechanisms acting on a flow field at once can be investigated. These capabilities set the toolbox apart from any other commercially available toolbox, and also surpass the capabilities of existing research codes.

On the scientific end, it was necessary to entirely redesign every portion of the traditional low dimensional model development cycle shown in Figure 8. This was necessary in order to obtain a low dimensional model that correctly captures the interaction of actuation and flow field. Once this was accomplished, we showed that the resulting model is not just valid for the flow data used for its derivation, but also for the feedback controlled flow state and for different Reynolds numbers. To the best of our knowledge, this is the first time that a flow model of such verified performance has been developed.

6 Outlook

Within this program, we have demonstrated the applicability of our modeling approach to the benchmark problem of a two dimensional circular cylinder wake at a Reynolds number of 100. In this section, we would like to discuss the applicability of the method to more complex flows. As opposed to many other derivations of low dimensional models, which start with assumptions of linearity or special boundary or initial conditions which typically apply only to a single specific flow field, no such assumptions are inherent in the DPOD-ANN-ARX approach developed within this program at the US Air Force Academy.

We have shown some preliminary results of the DPOD-ANN-ARX approach at a Reynolds number of Re=20,000, where the von Karman vortices break down into smaller and smaller turbulent structures that ultimately dissipate their energy into heat. These smaller structures can be quite energetic and thus more and more POD modes will need to be retained in order to model a given fraction of the overall flow energy content. This behaviour of POD is due to the energy optimality of the procedure, and DPOD inherits this property from POD. As a result, both POD and DPOD models will become inherently large for flows that break down into turbulence. If the purpose of model development is feedback control, however, there may not be a need to model the small turbulent eddies in order to capture the dynamic behaviour of the large vortical structures. In the case of the circular cylinder wake, one may only be interested in modeling the von Karman type shedding for the reasons outlined in the previous paragraph. Thus, an approach where the flow data is subjected to either spatial or temporal filtering may be applied, as has been pursued by the authors with good preliminary results (Siegel et al. 2007). The approach proposed in this work removes small scale turbulent structures from the data used for POD mode derivation while retaining the large structures (i.e. von Karman vortices) that are of interest for feedback controller development. This approach is much like the use of spatial filtering in large eddy simulations employed in state of the art CFD solvers. With this approach, one introduces a choice of how much or how little of the smaller structures are included in the model. Thus, one can derive models with relatively few modes that nonetheless capture the dynamics of the flow that is relevant for feedback control. The DPOD-ANN-ARX approach is particularly suited to this type of modeling, since no turbulence model is required: As the entire model development is data driven and does not include projection onto the Navier Stokes equations, no closure equations are needed. The approach can thus be used as introduced here, with the only added step being a filtering process before the derivation of the DPOD modes. However, more detailed investigations into filter kernel type, size and cutoff wave length are needed.

An important question pertaining to the application of DPOD to flow fields with multiple equally dominant frequencies is the selection of appropriate bin boundaries. In the present investigation, higher frequency content was small in amplitude compared to the fundamental frequency of the vortex shedding, and thus the lift force with a simple peak detection algorithm was suitable for bin segmentation. If several dominant frequencies coexist, there are different possible approaches to segmentation. Using a phase accurate temporal notch filter as a preprocessing step, one may recover the fundamental frequency and determine bin boundaries in the same fashion as introduced in this work. Alternatively, it is conceivable to use open loop forcing to elevate the amplitude of one of the dominant frequencies at a time, thus allowing for discovery of the spatial flow features related to each frequency using multiple SPOD procedures, one for each of the frequencies of interest.

Having outlined possible pathways of how the DPOD-ANN-ARX approach may be applied and extended to flow fields at Reynolds number of technical interest, the question remains how applicable this approach may be to other flow geometries. We consider the circular cylinder wake a prototype flow featuring separated free shear layers that develop instabilities leading to vortex shedding. As such, there are similarities to many other flows of technical interest that contain free shear layers, featuring both simpler and more complex flow behavior. Examples that have been investigated by the authors are the separated flow over a stalled airfoil, free shear layers formed behind a D shaped cylinder, and the wake of an axisymmetric bluff body. While we yet have to apply the DPOD-ANN-ARX approach to these flows, we consider them promising candidates since they all feature large coherent structures resulting from instabilities. The interaction of these instabilities and their resulting structures with flow actuators of various kinds is of great technical interest, both for open and closed loop flow control. DPOD-ANN-ARX models may be used to investigate this interaction in a structured and quantitative fashion.

In summary, we developed this approach with the intent to use the resulting models for controller development in order to achieve control of the formation of large structures caused by flow instabilities. From a technical perspective, these types of flows are the most promising candidates for feedback flow control since instabilities can be influenced with relatively small amounts of actuation energy. This is important in the context of the power limitations inherent in state of the art of dynamic flow actuators. Our approach supports flow fields with many different modes present, and can also accommodate multiple actuator interaction allowing for MIMO control. We did not intend it to be used for random turbulent flows, but find that there are many technical applications where this limitation is of no importance.

With the basic tools necessary to develop useful feedback controllers in place, the USAFA team is looking forward to extending these tools to more complex flow fields by addressing the open issues outlined in the previous paragraphs. This has resulted in a follow on proposal which is currently being reviewed by AFOSR. At the same time, we are starting to apply the software tools developed in this program to applied problems of interest to the Air Force, in particular, the Aero-optic problems encountered when a laser beam passes through turbulent airflow encountered around a turret mounted on an aircraft. We strongly feel that feedback flow control can have a huge impact in this application.

References

Abergel, F. and Temam, R., "On some Control Problems in Fluid Mechanics", Theor. Comput. Fluid Dynamics, Vol. 1, 1990, p. 303.

Adrian, R.J., "On the Role of Conditional Averages in Turbulence Theory", *Proceedings of the Fourth Biennial Symposium on Turbulence in Liquids*, J. Zakin and G. Patterson (Eds.), Science Press, Princeton, 1977, pp. 323-332.

Asghar, A., and Jumper, E.J., "Phase Synchronization of Vortex Shedding from Multiple Cylinders Using Plasma Actuators", AIAA 2003-1028, 41st Aerospace Sciences Meeting and Exhibit, 6-9 January 2003, Reno NV

Ausseur, J. M., Pinier J., T., Glauser, M.N., Higuchi H., and Carlson, H., "Experimental Development of a Reduced-Order Model for Flow Separation Control", 44th AIAA Aerospace Meeting and Exhibit, 9-12 January 2006, Reno, Nevada, AIAA Paper 2006-1251.

Balas M.J., "Active Control of Flexible Systems", *Journal of Optimization Theory and Applications*, Vol. 25, No. 3, 1978, pp. 217-236.

Bayon de Noyer, "Tail Buffet Alleviation of High Performance Twin Tail Aircraft using Offset Piezoceramic Stack Actuators and Acceleration Feedback Control", Ph.D. Thesis, Aerospace Engineering, Georgia Institute of Technology, Atlanta, Georgia 30332, November 1999.

Blevins, R., "Flow Induced Vibration", 2nd Edition, Van Nostrand Reinhold, 1990, pp. 54-58.

Cameron, J., Sick, A., Cohen, K., Wetlesen, D., and Siegel, S., "Determination of an Effective Sensor Configuration for Suppression of the von Kármán Vortex Street", AIAA Region V Student Paper Conference, University of Colorado, Boulder, CO, April 9-11, 2003 .

Cattafesta III, L.N., Williams, D.R., Rowley, C.W., and Alvi, F.S., "Review of Active Control of Flow-Induced Cavity Resonance", AIAA Paper 2003-3567, June 2003.

Cohen K., Siegel S., McLaughlin T., and Gillies E., "Feedback Control of a Cylinder Wake Low-Dimensional Model", 55th APS/DFD Meeting, Vol. 47, No. 10, Paper DN 2, Dallas, TX, Nov. 24-26, 2002.

Cohen, K., Siegel, S., McLaughlin, T. & Gillies, E. 2003 Feedback Control of a Cylinder Wake Low-Dimensional Model. *AIAA Journal* **41**(7), 1389-1391.

Cohen, K., Siegel S., McLaughlin T., and Myatt J., "Proper Orthogonal Decomposition Modeling Of A Controllled Ginzburg-Landau Cylinder Wake Model", 41st AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV, Jan. 6-9 2003, Reston, VA: AIAA, Inc., AIAA Paper 2003-1292.

Cohen, K., Siegel S., McLaughlin T., and Myatt J., "Proper Orthogonal Decomposition Modeling Of A Controlled Ginzburg-Landau Cylinder Wake Model", 41st AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV, Jan. 6-9 2003, Reston, VA: AIAA, Inc., AIAA Paper 2003-1292.

Cohen, K., Siegel, S., Wetlesen, D., Cameron, J., and Sick, A., "Effective Sensor Placements for the Estimation of Proper Orthogonal Decomposition Mode Coefficients in von Kármán Vortex Street", *Journal of Vibration and Control*, Vol. 10, No. 12, 1857-1880, December 2004.

Cohen, K., Siegel S., and McLaughlin T.,"A Heuristic Approach to Effective Sensor Placement for Modeling of a Cylinder Wake", *Computers and Fluids,* Volume 35, Issue 1, January 2006, pp. 103-120.

Cohen, K., Siegel, S., Seidel, J., & McLaughlin, T. 2006 System Identification of a Low Dimensional Model of a Cylinder Wake. AIAA Paper 2006-1411.

Cohen, K., Siegel, S., Seidel, J., and McLaughlin, T.," "Reduced Order Modeling for Closed-Loop Control of Three Dimensional Wakes", Invited lecture at Invited Session organized by AIAA Fluid Dynamics TC, Emerging Experiments Discussion Group, 3rd AIAA Flow Control Conference, San Francisco, California, 5 - 8 June 2006, AIAA-2006-3356.

Cohen, K., Siegel, S., Seidel, J., and McLaughlin, T., "Neural Network Estimator for Low-Dimensional Modeling of a Cylinder Wake", 3rd AIAA Flow Control Conference, San Francisco, California, 5 - 8 June 2006, AIAA-2006-3491.

Constantinescu, G.S., Chapelet, M. and Squires, K.D., "Prediction of the turbulent flow over a sphere", accepted for publication in *AIAA Journal*, 2003.

Constantinescu, G.S., Pacheco, R. and Squires, K.D., "Detached-eddy simulation of flow over a sphere", *AIAA Paper 2002-0425*, 2002.

Corke, T.C. and Matlis, E., "Phased Plasma Arrays for Unsteady Flow Control", AIAA 2000-2323, Fluids 2000 Conference, 19-22 June 2000, Denver CO.

Corke, T.C., Jumper, E.J., Post, M.L., Orlov, D., and McLaughlin, T.E., "Application of Weakly-Ionized Plasmas as Wing Flow Control Devices", AIAA 2002-0350, 40th Aerospace Sciences Meeting and Exhibit, 7-10 January 2002.

Cybenko, G.V. 1989 Approximation by Superpositions of a Sigmoidal function. *Mathematics of Control, Signals and Systems* **2**, 303-314.

Deane, A.E., Kevrekidis, I.G., Karniadakis, G.E. & Orszag, S.A. 1991 Low-dimensional models for complex geometry flows: Application to grooved channels and circular cylinders. *Phys. Fluids* **3**(10).

Debiasi, M., Little, J, Caraballo, E., Yuan, X., Serrani, A., Myatt, J.H., and Samimy, M., "Influence of Stochastic Estimation on the Control of Subsonic Cavity Flow – A Preliminary Study", 3rd AIAA Flow Control Conference, San Francisco, California, 5 - 8 June 2006, AIAA-2006-3492

Enloe, C.L, McLaughlin, T.E., VanDyken, R.D., Kachner, K.D., Jumper, E.J., and Corke, T.C., "Mechanisms and Responses of a Single Dielectric Barrier Plasma", AIAA 2003-1021, 41st Aerospace Sciences Meeting and Exhibit, 6-9 January 2003, Reno NV

Forsythe, J.R., Hoffmann, K.A., Cummings, R.M., Squires, K.D.., "Detached-Eddy Simulation with Compressibility Corrections Applied to a Supersonic Axisymmetric Base," Journal of Fluids Engineering, Vol. 124, No. 4, 2002, pp. 911-923.

Forsythe, J.R., Squires, K.D., Wurtzler, K.E. and Spalart, P.R., "Detached-Eddy Simulation of Fighter Aircraft at High Alpha", AIAA 2002-0591, January 2002.

Forsythe, J.R., Strang, W., Hoffmann, K.A., "Validation of Several Reynolds-Averaged Turbulence Models in a 3D Unstructured Grid Code,"" AIAA 00-2552, June 2000.

Forsythe, J.R., Woodson, S.H., "Unsteady CFD Calculations of Abrupt Wing Stall Using Detached-Eddy Simulation", AIAA 2003-0594, Jan 2003.

Forsythe, J.R., Squires, K.D., Wurtzler, K.E. and Spalart, P.R., "Detached-Eddy Simulation of the F-15E at High Alpha," *AIAA Journal of Aircraft*, Vol. 41, No. 2, 2004, pp. 193-200.

Galletti, B., Bruneau, C.H., Zannetti, L. & Iollo, A. 2004 Low-order modelling of laminar flow regimes past a confined square cylinder. *J. Fluid Mech.* **503.**

Gerhard, J., Pastoor, M., King, R., Noack, B.R., Dillmann, A., Morzynski, M. & Tadmor, G. 2003 Model Based Control of Vortex Shedding using Low-Dimensional Galerkin Models. 33rd AIAA Fluid Dynamics Conference, AIAA 2003-4261

Gillies, E. A., "Low-dimensional Control of the Circular Cylinder Wake", *Journal of Fluid Mechanics*, Vol. 371, 1998, pp. 157-178.

Gillies, E. A. 1995 Low-Dimensional Characterization and Control of Non-Linear Wake Flows, Ph.D. Dissertation, Faculty of Engineering, Univ. of Glasgow, Glasgow, Scotland, U.K.

Glezer, A., Kadioglu, Z., Pearlstein, A.J., 1989 Development of an extended proper orthogonal decomposition and its application to a time periodically forced plane mixing layer, Phys. Fluids A 1 (8).

Hansen, R.P. and Forsythe, J.R, "Large and Detached Eddy Simulations of a Circular Cylinder Using Unstructured Grids", *AIAA Paper 2003-0775*, 2003.

Holmes, P., Lumley, J.L., and Berkooz, G., "Turbulence, Coherent Structures, Dynamical Systems and Symmetry", Cambridge University Press, Cambridge, 1996.

Karypis, G., Schloegel, K., and Kumar, V, ParMETIS: Parallel Graph Partitioning and Sparse Matrix Ordering Library Version 1.0. University of Minnesota, Department of Computer Science, Minneapolis, MN 55455, July 1997.

Koopmann, G. 1967 The Vortex Wakes of Vibrating Cylinders at Low Reynolds Numbers. *J. Fluid Mech.* **28** *Part 3*, 501-512

List, J., Byerley, A.R., McLaughlin, T.E., and VanDyken, R., "Using Plasma Actuators Flaps to Control Laminar Separation on Turbine Blades in a Linear Cascade", AIAA 2003-1026, 41st Aerospace Sciences Meeting and Exhibit, 6-9 January 2003, Reno NV.

Ljung, L. 1999 System Identification – Theory for the User. 2nd Edition, Prentice Hall, Upper Saddle River, N.J.

Ma, X.; Karniadakis, G. 2002 A low-dimensional model for simulating three-dimensional cylinder flow. *J. Fluid Mech.* **458.**

Macheret, S.O., Ionikh, Y.Z., Chernysheva, N.V., Yalin, A.P., Martinelli, L., and Miles, R.B., "Shock Wave Propogation and Siepersion in Glow Discharge Plasmas", Physics of Fluids 13 No. 9, September 2001, pp 2693-2705.

Macheret, S.O., Schneider, M.N., and Miles, R.B., "Magnetohydrodynamic and Electrohydrodynamic Control of Hypersonic Flows of Weakly Ionized Plasmas", 33rd AIAA Plasmadynamics and Lasers Conference, 20-23 May 2002, Maui, HI.

McLaughlin, T, Munska, M., Vaeth, J., Dauwalter, T., and Goode, J., "Plasma-Based Actuators for Cylinder Wake Vortex Control", AIAA 2004-2129, AIAA 2nd Flow Control Conference, June 2004, Portland, OR.

Min, C., Choi, H. 1999 Suboptimal Feedback Control of Vortex Shedding at Low Reynolds Numbers. *J. Fluid Mech.* **401**, 123-156.

Meirovitch, L., "Dynamics and Control of Structures", John Wiley & Sons, Inc., New York, 1990, pp. 313-351. Noack, B.R., Afanasiev, K., Morzynski, M. & Thiele, F. 2003 A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *J. Fluid Mech.* **497**, 335–363.

Morton, S.A., Steenman, M.B., Cummings, R.M. and Forsythe, J.R., "DES grid resolution issues for vortical flows on a delta wing and an F-18C", *AIAA 2003-1103*, 2003.

Morton, S.A., Cummings, R.M., Kholodar, D.B., "High Resolution Turbulence Treatment of F/A-18 Tail Buffet", AIAA *2004-1676*, 2004.

Murray, N.E., and Ukeiley, L.S., "Estimating the Shear Layer Velocity Field Above an Open Cavity from Surface Pressure Measurements", 32nd Fluid Dynamics Conference and Exhibit, 24-26 June 2002, St. Louis, MO, AIAA Paper 2002-2866.

Nelles, O. 2001 Nonlinear System Identification. Springer-Verlag Berlin Heidelberg.

Noack, B.R., Afanasiev, K., Morzynski, M. & Thiele, F. 2003 A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *J. Fluid Mech.* **497**, 335–363.

Noack, B.R., Tadmor, G. & Morzynski, M. 2004b Actuation models and dissipative control in empirical Galerkin models of fluid flows. American Control Conference, Boston, MA, U.S.A., Paper FrP15.6

Nørgaard, M., Ravn, O., Poulsen, N. K., and Hansen, L. K., "Neural Networks for Modeling and Control of Dynamic Systems", Springer-Verlag, London, UK, 2000.Oertel, H. Jr. 1990 Wakes Behind Blunt Bodies. *Annual Review of Fluid Mechanics* **22**, 539-564.

Panton, R. L. 1996 Incompressible Flow, 2nd Edition. New York: John Wiley & Sons.

Post, M.L., and Corke, T.C., "Separation Control of High Angle of Attack Airfoil Using Plasma Actuators", AIAA 2003-1024, 41st Aerospace Sciences Meeting and Exhibit, 6-9 January 2003, Reno NV

Rempfer, D. 2000 On Low-Dimensional Galerkin Models for Fluid Flow. *Theoretical and Comp. Fluid Mechanics* **14**, Number 2

Roshko, A, "Experiments on the Flow past a Circular Cylinder at Very High Reynolds Number", Journal of Fluid Mechanics 10(3), 345-356, 1961.

Roth, J.R., Sherman, D.M., Wilkinson, S.P., "Electrohydrodynamic Flow Control with a Glow-Discharge Surface Plasma", AIAA J. 38 No. 7, July 2000, p1166-1172

Roth, J.R., Sherman, D.M., Wilkinson, S.P., Boundary Layer Flow Control with a One Atmosphere Uniform Glow Discharge Surface Plasma", AIAA 98-0328, 36th Aerospace Sciences Meeting and Exhibit, January 12-15, 1998, Reno NV

Roth, J.R., Sin, H., Chandra, R., Madhan, M., "Flow Re-Attachment and Acceleration by Paraelectric and Peristaltic Electrohydrodynamic (EHD) Effects", AIAA 2003-0531, 41st Aerospace Sciences Meeting and Exhibit, 6-9 January 2003, Reno NV

Sakamoto, Haniu, 1990, "A study on vortex shedding from spheres in a uniform flow", J. Fluids Eng., 112, pp. 386-393.

Schneider, M.N., Macheret, S.O., and Miles, R.B., " Nonequilibrium Magnetohydrodynamic Control of Scramjet Inlets", AIA 2002-2251, 33rd AIAA Plasmadynamics and Lasers Conference, 20-23 May 2002, Maui, HI.

Schneider, M.N., Macheret, S.O., and Miles, R.B., "Comparative Analysis of MHD and Plasma Methods fo Scramjet Inlet Control", AIAA 2003-0170, 41st Aerospace Sciences Meeting and Exhibit, 6-9 January 2003, Reno NV.

Schwarz, V., Bestek, H., Fasel, H. 1994, "Numerical simulation of nonlinear waves in the wake of an axisymmetric bluff body", AIAA 94-2285.

Serrano, M., Leigh, E., Johnson III, W., Forsythe, J.R., Morton, S.A. and Squires, K.D., "Computational aerodynamics of the C-130 in airdrop configurations", *AIAA Paper 2003-0229*, 2003.

Siegel S., "Completion and Optimization of an Open Water Channel and Investigation of the Wake behind an Axisymmetric Bluff Body", Diploma Thesis, Universität Stuttgart, 1994.

Siegel S., "Experimental Investigation of the Wake behind an Axisymmetric Bluff Body", Ph.D. dissertation, University of Arizona, 1999.

Siegel S., Cohen K., McLaughlin T., and Myatt J., "Real-Time Particle Image Velocimetry for Closed-Loop Flow Control Studies", 41st AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV, Jan. 6-9 2003, Reston, VA: AIAA, Inc., AIAA Paper 2003-0920.

Siegel S., Cohen K., Smith D., and McLaughlin T, "Observability Conditions for POD Modes in a Circular Cylinder Wake", 55th APS/DFD Meeting, Vol. 47, No. 10, Paper DN 4, Dallas, TX, Nov. 24-26, 2002.

Siegel S., Cohen, K. and McLaughlin T., (2003a)"Feedback Control of a Circular Cylinder Wake in Experiment and Simulation", Invited lecture at the 33rd AIAA Fluid Dynamics Conference and Exhibit, Orlando, AIAA-2003-3569, June 23-26 2003.

Siegel S., Cohen, K., and McLaughlin T., (2003b), "Low-Dimensional Feedback Control of the von Karman Vortex Street at a Reynolds number of 100", (accepted for presentation at IUTAM Symposium: Fluid-Structure Interactions, Piscataway, USA, June 2-6 2003).

Siegel, S., Cohen, K. & McLaughlin, T. 2006 Numerical Simulations of a Feedback Controlled Circular Cylinder Wake. *AIAA Journal* vol. 44 no. 6.

Sirovich, L., "Turbulence and the Dynamics of Coherent Structures Part I: Coherent Structures", *Quarterly of Applied Mathematics*, Vol. 45, No. 3, 1987, pp. 561-571.

Smith D., Cohen K., Siegel S., McLaughlin T., "Towards Closed-loop Control of the Wake behind a Circular Cylinder Using Rotational Oscillations", (currently under review of the *Physics of Fluids*), 2003.

Smith, D. R., Siegel, S., and McLaughlin T., "Modeling of the wake Behind a Circular Cylinder Undergoing Rotational Oscillation", AIAA Paper 2002-3066, June 2002.

Spalart, P.R., "Strategies for Turbulence Modeling and Simulations", *International Journal of Heat and Fluid Flow*, **21**, p. 252-263, 2000.

Spalart, P.R., Jou, W-H., Strelets, M. , and Allmaras, S.R., "Comments on the Feasibility of LES for Wings, and on a Hybrid RANS/LES Approach," *Advances in DNS/LES*, 1st AFOSR International Conference on DNS/LES, Greyden Press, Columbus, OH, 1997.

Squires, K.D., Forsythe, J.R. and Spalart, P.R., "Detached-Eddy Simulation of the separated flow around a forebody cross-section", *Direct and Large-Eddy Simulation IV*, ERCOFTAC Series – Volume 8, B.J. Geurts, R. Friedrich and O. Metais, editors, Kluwer Academic Press, pp. 481-500, 2001.

Squires, K.D., Forsythe, J.R., Morton, S.A., Strang, W.Z., Wurtzler, K.E., Tomaro, R.F., Grismer, M.J. and P.R. Spalart, "Progress on Detached-Eddy Simulation of massively separated flows", *AIAA Paper 2002-1021*, 2002.

Strang, W.Z., Tomaro, R.F, Grismer, M.J., 1999, "The Defining Methods of Cobalt60: a Parallel, Implicit, Unstructured Euler/Navier-Stokes Flow Solver", AIAA 99-0786, January 1999.

Strelets, M., "Detached Eddy Simulation of Massively Separated Flows," *AIAA Paper 2001-0879*, 2001.

Stuart, J.T. 1958 On the non-linear mechanics of hydrodynamic stability. *J. Fluid Mech.* **4**.

Tomaro, R.F., Strang, W.Z., and Sankar, L.N., 1997, ``An Implicit Algorithm for Solving Time Dependent Flows on Unstructured Grids, AIAA 97-0333, January 1997.

Travin, A., Shur, M., Strelets, M., and Spalart P., "Detached-Eddy Simulations Past a Circular Cylinder", Flow Turbulence and Combustion, 63, 293-313, 1999.

van Nunen, J.W.G., "Pressure and Forces on a Circular Cylinder in a Cross Flow at High Reynolds Numbers", in Naudasher, E. (ed.), Flow Induced Structural Vibrations, Springer-Verlag, Berlin, 748-754, 1974.

Williamson, C. H. K. 1996 Vortex Dynamics in the Cylinder Wake. *Annual Review of Fluid Mechanics* **28**, 477-539.

Publications within this Program

Journal Articles and other peer reviewed work

Cohen, K.; Siegel, S.; McLaughlin, T.; Gillies, E.; Myatt, J., 'Closed-loop approaches to control of a wake flow modeled by the Ginzburg-Landau equation', Computers and Fluids, Vol. 34 Issue 8, September 2005

Stefan G. Siegel, Kelly Cohen, Thomas McLaughlin, 'Numerical Simulations of a Feedback Controlled Circular Cylinder Wake', AIAA Journal, vol. 44 no. 6, June 2006

Cohen, K.; Siegel, S.; McLaughlin, T., 'A Heuristic Approach to Effective Sensor Placement for Modeling of a Cylinder Wake', Computers and Fluids, Vol. 35, 2006

Haiqian Yu; Miriam Leeser; Gilead Tadmor; Stefan Siegel, 'Real-Time Particle Image Velocimetry for Feedback Loops Using FPGA Implementation', Journal of Aerospace Computing, Information, and Communication , vol. 3 no. 2, 2006

Suzen, Y.B., Huang, P.G., Jacob, J.D., Ashpis, D.E., "Numerical Simulations of Plasma Based Flow Control Applications," AIAA 2005-4633, June 6-9, 2005.

Siegel, Stefan; Cohen, Kelly; Seidel, Jürgen; Luchtenburg, Mark; McLaughlin, Thomas, 'Low Dimensional Modeling of a Transient Cylinder Wake Using Double Proper Orthogonal Decomposition', accepted for publication in J. Fluid Mechanics, June 2007

Stefan Siegel, Kelly Cohen, Jürgen Seidel, Thomas McLaughlin, 'State estimation of transient flow fields using Double Proper Orthogonal Decomposition (DPOD)', Active Flow Control, NNFM 95, R. King (Editor), Springer, pp. 105-118, 2007

Seidel, Jürgen; Cohen, Kelly; Aradag, Selin; Siegel, Stefan; McLaughlin, Thomas, 'Reduced Order Modeling of a Turbulent Three Dimensional Cylinder Wake', to be submitted to Computers and Fluids, Summer 2007.

Kelly Cohen, Stefan Siegel, Jürgen Seidel, Selin Aradag, and Thomas McLaughlin. 'Reduced Order Model Based Controller Design for Feedback Flow Control', Abstract to be submitted to Prog. Aerospace Sci., Summer 2007

Kelly Cohen, Stefan Siegel, Jürgen Seidel, Selin Aradag, and Thomas McLaughlin, 'Sensor based estimation of POD Flow States using Artificial Neural Networks', to be submitted to Computers and Fluids, Summer 2007

Conference Presentations

S. Siegel, K. Cohen, J. Seidel, and T. McLaughlin, 'Proper Orthogonal Decomposition Snapshot Selection for State Estimation of Feedback Controlled Flows', 44th AIAA Aerospace Sciences Meeting and Exhibit, AIAA-2006-1400, 2006

J. Seidel, S. Siegel, K. Cohen, V. Becker, and T. McLaughlin, 'Simulations of Three Dimensional Feedback Control of a Circular Cylinder Wake', 44th AIAA Aerospace Sciences Meeting and Exhibit, AIAA-2006-1404, 2006

K. Cohen, S. Siegel, J. Seidel, and T. McLaughlin, 'System Identification of a Low Dimensional Model of a Cylinder Wake', 44th AIAA Aerospace Sciences Meeting and Exhibit, AIAA-2006-1411, 2006

J. Seidel, S. Siegel, K. Cohen,and T. McLaughlin, 'Simulations of Flow Control of the Wake Behind an Axisymmetric Bluff Body', 3rd AIAA Flow Control Conference San Francisco, 5-8th June 2006, AIAA 2006-3490, 2006

K. Cohen, S. Siegel, J. Seidel, and T. McLaughlin, 'Reduced Order Modeling for Closed- Loop Control of Three Dimensional Wakes', 3rd AIAA Flow Control Conference San Francisco, 5-8th June 2006, AIAA-2006-3356, 2006

K. Cohen, S. Siegel, J. Seidel, and T. McLaughlin, 'Neural Network Estimator for Low-Dimensional Modeling of a Cylinder Wake', 3rd AIAA Flow Control Conference San Francisco, 5-8th June 2006, AIAA-2006-3491, 2006

Shin, Young-Sug, Cohen, K., Siegel, S., Seidel, J., and McLaughlin, T., "Neural Network Estimator for Closed-Loop Control of a Cylinder Wake", AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, Colorado, 21 - 24 Aug 2006, AIAA-2006-6428.

Stefan Siegel, Kelly Cohen, Jürgen Seidel, Thomas McLaughlin, 'State estimation of transient flow fields using Double Proper Orthogonal Decomposition (DPOD)', Conference on Active Flow Control, Berlin, Sept 27-29th, 2006

Siegel, Stefan; Cohen, Kelly; Seidel, Jürgen; Aradag, Selin; McLaughlin, Thomas, 'Low Dimensional Modeling of Transient Flow Fields Using Double Proper Orthogonal Decomposition', 59th Annual APS / DFD Meeting , Tampa, Fl , 2006

Aradag, Selin; Cohen, Kelly; Seidel, Jürgen; Siegel, Stefan; McLaughlin, Thomas, 'Large Eddy Simulations of Flow Over A Circular Cylinder Using Unstructured Grids', 59th Annual APS / DFD Meeting , Tampa, Fl, 2006

David R. Williams, Kelly Cohen, Stefan Siegel, Tom McLaughlin, 'Open-loop and closed-loop excitation of the wake behind a circular cylinder.', 59th Annual APS / DFD Meeting , Tampa, Fl, 2006

Cohen, K., Siegel, S., Seidel, J., and McLaughlin, T., "Low dimensional modeling, estimation and control of a cylinder wake", Invited lecture at Invited Session organized by Dr. James Myatt of AFRL/VA called "Order Reduction and Control for Aerodynamics", 45th IEEE Conference on Decision and Control, San Diego, 13-15 December 2006.

Siegel, Stefan; Aradag, Selin; Seidel, Jürgen; Cohen, Kelly; McLaughlin, Thomas, 'Low Dimensional POD based Estimation of a 3D Turbulent Separated Flow', 45th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, AIAA-2007-0112

Young-Sug Shin, Kelly Cohen, Stefan Siegel, Jürgen Seidel and Thomas McLaughlin, 'Neural Network Estimation of Transient Flow Fields using Double Proper Orthogonal Decomposition', 45th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, AIAA-2007-1166

Siegel, Stefan; Cohen, Kelly; Seidel, Jürgen; Aradag, Selin; McLaughlin, Thomas, 'Simulation of a Feedback-Controlled Cylinder Wake Using Double Proper Orthogonal Decomposition', 37th AIAA Fluid Dynamics Conference , AIAA Paper-2007-4502, Miami, FL, 2007

Seidel, Jürgen; Cohen, Kelly; Aradag, Selin; Siegel, Stefan; McLaughlin, Thomas, 'Reduced Order Modeling of a Turbulent Three Dimensional Cylinder Wake', 37th AIAA Fluid Dynamics Conference , AIAA Paper-2007-4503, Miami, FL, 2007

Cohen, K., Siegel, S., Seidel, J., Aradag, S., and McLaughlin, T., "Nonlinear Estimation of Transient Flow Field Low Dimensional States Using Double Proper Orthogonal Decomposition", AIAA-2007-2836, AIAA Infotech, Sonoma, CA, May 7-10, 2007.

Seidel, J., Siegel, S., Cohen, K., and McLaughlin, T., "Feedback Control of a Circular Cylinder Wake", 3rd International Symposium on Integrating CFD and Experiments in Aerodynamics, USAF Academy, CO, June 20-21, 2007.

Kelly Cohen, Stefan Siegel, Jürgen Seidel, Selin Aradag, and Thomas McLaughlin, 'Reduced Order Model Based Controller Design for Feedback-Controlled Cylinder Wake', Abstract accepted for the 46th Aerospace Sciences Meeting and Exhibit, Reno, NV, 2008

Jürgen Seidel, Stefan Siegel, Kelly Cohen, Selin Aradag, and Thomas McLaughlin, 'Data Analysis of an Axisymmetric Bluff Body Wake using Fourier Transform and POD', Abstract accepted for the 46th Aerospace Sciences Meeting and Exhibit, Reno, NV, 2008

Selin Aradag, Stefan Siegel, Jürgen Seidel, Kelly Cohen, and Thomas McLaughlin, 'Filtered POD based Estimation of 3D Turbulent Separated Flows', Abstract accepted for the 46th Aerospace Sciences Meeting and Exhibit, Reno, NV, 2008

Stefan Siegel, Jürgen Seidel, Kelly Cohen, Selin Aradag, and Thomas McLaughlin, 'Open Loop Transient Forcing of an Axisymmetric Bluff Body Wake', Abstract accepted for the 46th Aerospace Sciences Meeting and Exhibit, Reno, NV, 2008

Appendix A: Toolbox Manual and Tutorial

cobalt solutions

Cobalt
Feedback Flow Control Manual

Cobalt Beta 3.9

Cobalt Solutions, LLC.
4636 New Carlisle Pike
Springfield, OH 45504-3336

www.cobaltcfd.com
email: *contact@cobaltcfd.com* or *support@cobaltcfd.com*

cobalt solutions

Cobalt version 4.0

cobalt solutions

Software License Agreement for *Cobalt*

THIS SOFTWARE LICENSE AGREEMENT IS FOR THE SOFTWARE PRODUCTS ("SOFTWARE") NAMED **COBALT** ©2002-2007 by Cobalt Solutions, LLC, **LIGASE** ©2002-2007 by Cobalt Solutions, LLC, **BLACKSMITH** ©2002-2007 by Cobalt Solutions, LLC. THIS COBALT SOLUTIONS, LLC ("COBALT SOLUTIONS") SOFTWARE LICENSE AGREEMENT ("LICENSE AGREEMENT") IS A LEGAL AGREEMENT BETWEEN YOU (EITHER AN INDIVIDUAL OR A SINGLE ENTITY) AND COBALT SOLUTIONS, LLC FOR THE SOFTWARE PRODUCT IDENTIFIED ABOVE.

This Cobalt Solutions, LLC ("Cobalt Solutions") software license agreement ("License Agreement") is a legal agreement between you (either an individual or a single entity) and Cobalt Solutions, LLC for the software product identified above. By installing, copying or otherwise using the software product, you agree to be bound by the terms of this license agreement. If you do not agree to the terms of the license agreement, do not install or use the software product; you may, however, return it to your place of purchase for a full refund.

Software License Agreement

This Software is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. This Software is licensed, not sold.

For purposes of this License Agreement, "Software Product" refers to the computer software and associated media, printed materials, and "online" or electronic documentation, including without limitation any and all executable files, add-ons, filters, tutorials, help files and other files, that accompany the Cobalt Solutions' product identified above, herein or in the accompanying documentation; "Install" means to copy the Software Product to a hard disk drive or similar storage device; "Use" means loading the Software Product into computer memory or running it on a single or multiple central processing unit(s); and "You" and "Recipient" means the company, entity or individual whose funds are used to pay the license fee or who has otherwise acquired the Software Product; "Cobalt Solutions" means the entity granting this license to You under the terms and conditions of this License Agreement.

1. Grant of License

Software Product. Cobalt Solutions grants You a non-exclusive license to Use the Software Product and documentation subject to the restrictions and terms set forth in the License Agreement. If You have purchased a Single-User License, You may install and Use the Software Product on one computer; if You have purchased a Multiple-User License, You may install and Use the Software Product on the associated computer by the licensed number of users; if You have purchased a Network License You may install and Use the Software Product on any compatible computer on Your network by the licensed number of concurrent users. You may make a single copy of the Software Product as an archive copy, provided that it includes all notices and markings, including copyright, trademark, and other proprietary notices as on the original, and which may not be in Use at any time, unless the original is damaged beyond Use, and must remain in Your possession and control.

2. Copyright

The only right granted to You is the right to Use the Software Product and accompanying documentation in accordance with this License Agreement. All rights not expressly granted to You in the License Agreement are specifically reserved by Cobalt Solutions. You do not receive or acquire the right, title, or interest to the Software Product, or to any applicable patents, trademarks, copyrights or trade-secrets. You may not remove or alter any proprietary notices, labels, or trademarks on the Software Product or accompanying documentation. You may not modify, translate, copy, reproduce, reverse engineer, disassemble, decompile, or otherwise derive source code from, the Software Product or accompanying documentation, or use it as a basis for the preparation of other software programs or derivative works, or use it in any manner that infringes the intellectual property or other rights of Cobalt Solutions or another party, except as permitted herein or under applicable law. The Software Product and accompanying documentation may not be transmitted electronically, including over the Internet, rented, loaned, leased, sold, distributed, made available, directly or indirectly, for use by any other person or entity not covered by this License Agreement, used by third parties in a service bureau or otherwise transferred, transmitted or used without authorization under this License Agreement.

3. Description of Other Rights and Limitations

Academic Software Product. If the Software Product is identified as "Academic", You must be a Qualified Educational User (as defined by Cobalt Solutions) to Use the Software Product. If You are not a Qualified Educational User, You have no rights under this License Agreement. To determine whether you are a Qualified Educational User, please contact Cobalt Solutions at the applicable address set forth on the next page.

Support Services. Cobalt Solutions may provide You with technical support services related to the Software Product ("Support Services"). Use of Support Services is governed by the Cobalt Solutions policies and programs described in "online" or electronic documentation, and/or in other Cobalt Solutions-provided materials. Any supplemental software code provided to You as part of the Support Services shall be considered part of the Software Product and subject to the terms and conditions of this License Agreement. With respect to any technical information that You provide to Cobalt Solutions as part of the Support Services, Cobalt Solutions may use such information for its business purposes, including product support and development. Cobalt Solutions will not utilize such technical information in a manner that personally identifies You.

Termination. Unauthorized copying or duplication of the Software Product will result in automatic termination of this License Agreement. Without prejudice to any other rights, Cobalt Solutions may terminate this License Agreement if You fail to comply with the terms and conditions of the License Agreement. In such event, You must destroy all copies of the Software Product and all of its component parts.

Transfer. You may not rent, lease, sublicense, or lend the Software Product or documentation. You may not transfer the rights to use the Software Product to another person or legal entity.

4. Reports and Audit Rights
You shall institute reasonable measures to ensure compliance with the terms and conditions of this License Agreement. Upon Cobalt Solutions reasonable request, You agree to provide reports relating to Your Use of the Software Product as necessary to demonstrate compliance with the terms and conditions of the license Agreement. You further agree that Cobalt Solutions has the right, upon reasonable prior notice, to audit Your records and inspect Your facilities to verify compliance with the terms and conditions of this license Agreement.

5. U.S. Government Restricted Rights
The Software Product and documentation are provided with RESTRICTED RIGHTS. Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraphs (a) through (d) of the Commercial Computer-Restricted Rights clause at FAR 52.227-19 when applicable, or in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, and in similar or successor clauses in the DOD or NASA FAR Supplement. Contractor/manufacturer is Cobalt Solutions, LLC., 4636 New Carlisle Pike, Springfield, OH 45504-3336.

6. Export Restrictions
You may not export or reexport the Software Product or any underlying information or technology except in full compliance with all United States and other applicable laws and regulations. In particular, but without limitation, none of the Software Product or underlying information or technology may be exported or reexported (a) into (or to a national or resident of) Cuba, Haiti, Iran, Iraq, Libya, Serbia, Montenegro, North Korea, Sudan or Syria or (b) to anyone on the U.S Treasury Department's list of Specially Designated Nationals of the U. S. Commerce Department's Table of Deny Orders, as such countries, lists and orders may be amended or modified from time to time. By Using the Software Product, You are specifically agreeing to the foregoing and You are representing and warranting that You are not located in, under the control of, or a national or resident of any such country or any such list.

7. Software Updates
At Cobalt Solutions' sole discretion, Cobalt Solutions may provide You with updates to the Software Product. Cobalt Solutions retains the right to provide updates for a fee. You may refuse to accept updates. By accepting updates, You agree to destroy all prior copies and versions of the Software Product. The conditions outlined in this License Agreement are only transferred to the updated Software Product when You have fully complied with the preceding requirements.

8. Dual-Media Software
You may receive the Software Product in more than one medium. Regardless of the type or size of medium You receive, You may use only one medium that is appropriate for Your single computer. You may not loan, rent, lease, or otherwise transfer the other medium to another user, except as part of the Description of Transfer (as provided above) of the Software Product.

9. Governing Law
This License Agreement will be governed by the laws in force in the State of Ohio. You agree that the Clark County Common Pleas Court shall have jurisdiction and venue over any dispute arising out of this License Agreement and Your Use of this Software Product. The prevailing party shall be awarded reasonable costs for any legal proceedings to this License Agreement, including reasonable attorneys' fees and costs. This License Agreement will not be governed by the United Nations Convention on Contracts for the International Sale of Goods, the application which is expressly excluded.

10. Software Product Limited Warranty
Limited Warranty. Cobalt Solutions warrants that (a) the Software Product will perform substantially in accordance with the accompanying written materials for a period of ninety (90) days from the date of receipt, and (b) any Support Services provided by Cobalt Solutions shall be substantially as described in applicable written materials provided to You by Cobalt Solutions, and Cobalt Solutions support engineers will make commercially reasonable efforts to solve any problem issues. Some states and jurisdictions do not allow limitations on duration of any implied warranty, so the above limitation may not apply to You. To the extent allowed by applicable law, implied warranties on the Software Product, if any, are limited to ninety (90) days.

Customer Remedies. Cobalt Solutions' and its supplier's entire liability and Your exclusive remedy shall be, at Cobalt Solutions' option, either (a) return of the price paid, if any, or (b) repair or replacement of the Software Product that does not meet Cobalt Solutions' limited Warranty and which is returned to Cobalt Solutions with a copy of Your receipt. This Limited Warranty is void if failure of the Software Product has resulted from accident, abuse, or misapplication. Any replacement Software Product will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer. Outside the United States, neither these remedies nor any product Support Services offered by Cobalt Solutions are available without proof of purchase from an authorized international source.

NO OTHER WARRANTIES. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, COBALT SOLUTIONS AND ITS SUPPLIERS DISCLAIM ALL OTHER WARRANTIES AND CONDITIONS, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND NON-INFRINGEMENT, WITH REGARD TO THE SOFTWARE PRODUCT, AND THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES. THIS LIMITED WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS. YOU MAY HAVE OTHERS, WHICH VARY FROM STATE/JURISDICTION TO STATE/JURISDICTION.

LIMITATION OF LIABILITY. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL COBALT SOLUTIONS OR ITS SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR ANY OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE PRODUCT OR THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES, EVEN IF COBALT SOLUTIONS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN ANY CASE, COBALT SOLUTIONS' ENTIRE LIABILITY UNDER ANY PROVISION OF THIS LICENSE AGREEEMENT SHALL BE

LIMITED TO THE GREATER OF THE AMOUNT ACTUALLY PAID BY YOU FOR THE SOFTWARE PRODUCT OR U.S.$5.00; PROVIDED HOWEVER, IF YOU HAVE ENTERED INTO AN COBALT SOLUTIONS SUPPORT SERVICES AGREEMENT, COBALT SOLUTIONS' ENTIRE LIABILITY REGARDING SUPPORT SERVICES SHALL BE GOVERNED BY THE TERMS OF THAT AGREEMENT. BECAUSE SOME STATES AND JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY, THE ABOVE LIMITAION MAY NOT APPLY TO YOU.

11. Entire License Agreement
This License Agreement constitutes the entire agreement between parties pertaining to the subject matter hereof and supersedes all prior representations, warranties, conditions, agreements, and understandings, whether oral or written, express or implied, relating to this License Agreement. No supplement, modification, or waiver of this License Agreement shall be effective unless it is provided or approved by COBALT SOLUTIONS in writing.

Table of Contents

1.Installation

Take the provided tar file provided, and move it to your Cobalt directory, and untar it.

1. mv flowcontrol.tar.gz $CBLTLOC
2. cd $CBLTLOC
3. tar xvfz flowcontrol.tar.gz

Then run Matlab (on the machine you will be running Cobalt on), and add the $CBLTLOC/flowcontrol/matlab directory to your path including subdirectories

1. Under the File menu – click "SET PATH"
2. In the dialog that comes up, click "Add with Subfolders"
3. Navigate to $CBLTLOC/flowcontrol/matlab, select it, and click OK.
4. Click on Save

Then add the bin directory of the flowcontrol directory to your path ($CBLTLOC/flowcontrol/bin). This can be done if you use csh by adding to your .cshrc:

```
setenv PATH $CBLTLOC/flowcontrol/bin:$PATH
```

or with bash, add to your .bashrc file:

```
PATH=$CBLTLOC/flowcontrol/bin/:$PATH
export PATH
```

2.Cob alt/Matlab Interface

Cobalt uses the Matlab Engine library to directly call Matlab, and to pass variables to and from Matlab. This requires that Matlab be installed on at least the first processor that *Cobalt* is run on.

Passed Variables

The following is a list of the variables that are passed to and from *Cobalt* and Matlab. Do not use any variables that end in _CBLT within Matlab, as these are reserved for use by Cobalt. All datatypes are double.

<u>SensorData (Cobalt Output):</u>
This array contains the information from all the sensors that were requested. NT is the number of total sensors.
SensorData(1,NT): X-coordinate of sensor NT
SensorData (2,NT): Y-coordinate of sensor NT
SensorData (3,NT): Z-coordinate of sensor NT
SensorData (4,NT): density at sensor NT
SensorData (5,NT): X-component of velocity at sensor NT
SensorData 6,NT): Y-component of velocity at sensor NT
SensorData (7,NT): Z-component of velocity at sensor NT
SensorData (8,NT): pressure at sensor NT

<u>GlobalData (Cobalt Output):</u>
This array contains global data including the forces an moments on the grid.
GlobalData(1): X-component of force
GlobalData (2): Y-component of force
GlobalData (3): Z-component of force
GlobalData (4): X-component of moment
GlobalData (5): Y-component of moment
GlobalData (6): Z-component of moment
GlobalData (7): time at beginning of time-step
GlobalData (8): time at end of time-step
GlobalData((9): Iteration Number. Zero if prior to first iteration.

BasisData (Cobalt Output):
This array contains the transformation matrix of the grid motion with respect to the lab reference frame.
BasisData (1,1): component of X^ in X-direction at start of time-step
BasisData (2,1): component of X^ in Y-direction at start of time-step
BasisData (3,1): component of X^ in Z-direction at start of time-step
BasisData (1,2): component of Y^ in X-direction at start of time-step
BasisData (2,2): component of Y^ in X-direction at start of time-step
BasisData (3,2): component of Y^ in X-direction at start of time-step
BasisData (1,3): component of Z^ in X-direction at start of time-step
BasisData (2,3): component of Z^ in X-direction at start of time-step
BasisData (3,3): component of Z^ in X-direction at start of time-step

RBM_Data (Cobalt Input):
This array contains the new transformation matrix determined by the controller (in the case of motion control), as well as the translation information.
RBM_Data (1): component of X^ in X-direction at end of time-step
RBM_Data (2): component of X^ in Y-direction at end of time-step
RBM_Data (3): component of X^ in Z-direction at end of time-step
RBM_Data (4): component of Y^ in X-direction at end of time-step
RBM_Data (5): component of Y^ in Y-direction at end of time-step
RBM_Data (6): component of Y^ in Z-direction at end of time-step
RBM_Data (7): component of Z^ in X-direction at end of time-step
RBM_Data (8): component of Z^ in Y-direction at end of time-step
RBM_Data (9): component of Z^ in Z-direction at end of time-step
RBM_Data (10): X-coordinate of origin of X^Y^Z^ system at beginning of time-step
RBM_Data (11): Y-coordinate of origin of X^Y^Z^ system at beginning of time-step
RBM_Data (12): Z-coordinate of origin of X^Y^Z^ system at beginning of time-step
RBM_Data (13): X-component of displacement of origin of X^Y^Z^ system over time-step
RBM_Data (14): Y-component of displacement of origin of X^Y^Z^ system over time-step
RBM_Data (15): Z-component of displacement of origin of X^Y^Z^ system over time-step

BC_Data (Cobalt Input):
For blowing/suction control of boundary conditions, this array contains the control information. N is the number of control boundary conditions. The meaning of the array depends on whether the boundary condition was mass flow or velocity based.
Mass Flow BC
BC_Data (1,N): Mass Flow
BC_Data (2,N): Total Pressure
BC_Data (3,N): Total Temperature
Velocity BC
BC_Data (1,N): Velocity Magnitude
BC_Data (2,N): Static Density
BC_Data (3,N): Temporal Derivative of Velocity

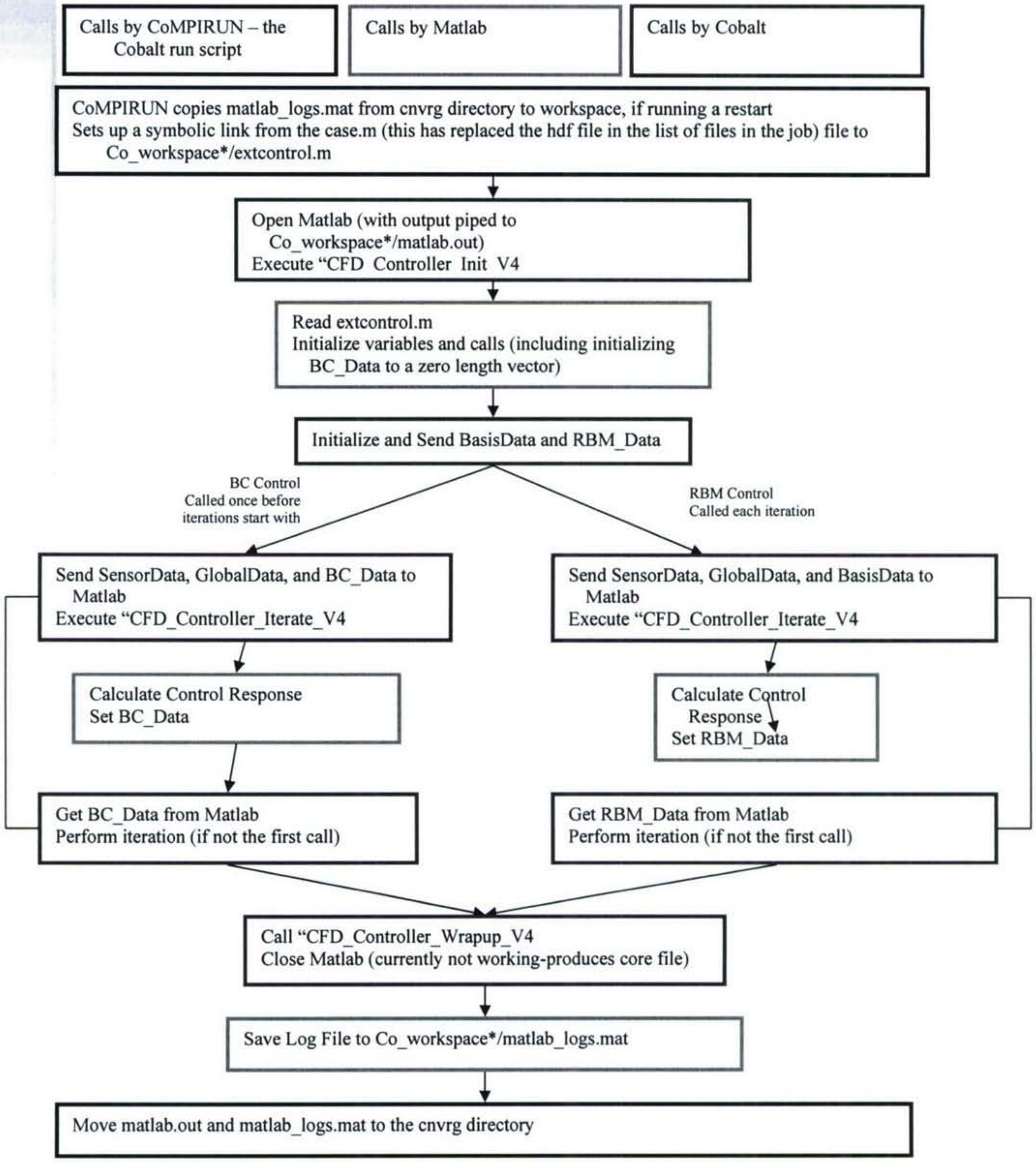
<u>Body Force Data (Cobalt Input):</u>
For prescribed body forces, Matlab must specify a number of actuators, and a structured grid for each actuator. Various information must be supplied as described below.
N_BF_Actuators(1): Number of Body Force actuators (may change each iteration)
GridDims(3,N_BF_Actuators): Dimensions of body force grids for each actuator (IMAX,JMAX,KMAX)
XBF{N_BF_Actuators,1}: Cell with entries for each actuator that contains the body force grid coordinates. Each entry described below.
XBF{N,1}: Array with dimensions (3,GridDims(1,N),GridDims(2,N),GridDims(3,N)) – coordinates for actuator n. Coordinate system must be right-handed.
BF{N_BF_Actuators,1}: Cell with entries for each actuator that contains the body force grid body forces. Each entry described below.
XBF{N,1}: Array with dimensions (3,GridDims(1,N),GridDims(2,N),GridDims(3,N)) – body force for actuator N

Body Force notes:
A Matlab code fragment is listed below that will set up two actuators.

```
N_BF_Actuators=2;
XBF=cell(N_BF_Actuators,1);
BF=cell(N_BF_Actuators,1);
GridDims=zeros(3,N_BF_Actuators);
GridDims(:,1)=[10 10 10 ];
GridDims(:,2)=[20 20 20 ];
for nact=1:N_BF_Actuators
                    XBF_tmp=zeros(3,GridDims(1,nact),GridDims(2,nact),GridDims(3,nact));
       BF_tmp=zeros(3,GridDims(1,nact),GridDims(2,nact),GridDims(3,nact));
       for i=1:GridDims(1,nact)
                         for j=1:GridDims(2,nact)
                         for k=1:GridDims(3,nact)
                         XBF_tmp(1,i,j,k)=(double(i)-1)/double(GridDims(1,nact)-1)+nact-1.5;
                               XBF_tmp(2,i,j,k)=(double(j)-1)/double(GridDims(2,nact)-1);
                               XBF_tmp(3,i,j,k)=(double(k)-1)/double(GridDims(3,nact)-1);
                               BF_tmp(1,i,j,k)=(double(i)-1)/double(GridDims(1,nact)-1);
                         BF_tmp(2,i,j,k)=(double(j)-1)/double(GridDims(2,nact)-1);
                         BF_tmp(3,i,j,k)=(double(k)-1)/double(GridDims(3,nact)-1);
                         end
                         end
                   end
                   XBF{nact}=XBF_tmp;
                   BF{nact}=BF_tmp;
                   clear XBF_tmp;
                   clear BF_tmp;
 end
```

Interface Flowchart

A flowchart is on the following page that describes the interface between Cobalt and Matlab. The body force controller is not listed, although it gets called each iteration as the other controllers do. But it currently does not allow for sensor data. In the release version, sensors for all flow control types will be consolidated so that they can be used for any controller type.


Calls by CoMPIRUN – the Cobalt run script
Calls by Matlab
Calls by Cobalt
CoMPIRUN copies matlab_logs.mat from cnvrg directory to workspace, if running a restart
Sets up a symbolic link from the case.m (this has replaced the hdf file in the list of files in the job) file to Co_workspace*/extcontrol.m
Open Matlab (with output piped to Co_workspace*/matlab.out)
Execute "CFD_Controller_Init_V4
Read extcontrol.m
Initialize variables and calls (including initializing BC_Data to a zero length vector)
Initialize and Send BasisData and RBM_Data
BC Control Called once before iterations start with
RBM Control Called each iteration
Send SensorData, GlobalData, and BC_Data to Matlab
Execute "CFD_Controller_Iterate_V4
Send SensorData, GlobalData, and BasisData to Matlab
Execute "CFD_Controller_Iterate_V4
Calculate Control Response
Set BC_Data
Calculate Control Response
Set RBM_Data
Get BC_Data from Matlab
Perform iteration (if not the first call)
Get RBM_Data from Matlab
Perform iteration (if not the first call)
Call "CFD_Controller_Wrapup_V4
Close Matlab (currently not working-produces core file)
Save Log File to Co_workspace*/matlab_logs.mat
Move matlab.out and matlab_logs.mat to the cnvrg directory

Matlab Controller Routines

Listed below are the main Matlab subroutines used. The list is not comprehensive, and you will be developing your own subroutines to interface with these. But these provide a starting point, and an overall shell for performing closed loop flow control.

Routines Directly Called by Cobalt

CFD_Controller_Init_V4.m

- Used to initialize a number of variables needed during the feedback control run.
- Reads the matlab file provided by the user for variables.
- Calls CFD_Controller_Calls_V4.m to set up the calls for the mode filter, controller, filter, and output modules.
- *No user modification should be necessary for standard feedback control runs.*

CFD_Controller_Calls_V4.m

- Sets up the Matlab function calls for the mode filter, controller, filter, and output modules.
- *No user modification should be necessary for standard feedback control runs.*

CFD_Controller_Iterate_V4.m

- Main feedback control iteration routine called at every time step from Cobalt.
- First, the state of the simulation is checked. If it is a restart, the log file is read. If the simulation terminated abnormally and the time in the log file does not match the time of the restart file, the log variables are cut to match the restart time.
- The routine calls the mode filter and the controller, and then writes the controller output into the log variables. Next, the filter is called and the output is added to the log. Finally, the output module is called and the log data structure is written to the log file.
- *No user modification should be necessary for standard feedback control runs.*

CFD_Controller_Wrapup_V4.m

- Save the controller log file and print diagnostic information.
- *No user modification should be necessary for standard feedback control runs.*

Routines Called within Matlab

Mode Filter

- The mode filter routine is used to filter the sensor information provided by Cobalt and convert it to POD mode amplitudes which are used to compute the control input.
- *User modification is required for the implementation of the desired filter functionality.*

Controller

- Compute the controller signal based on the POD mode amplitudes provided by the mode filter. Uses the Cont data structure provided in the $CASE.m file for specification of various controller settings.
- *User modification is required for the implementation of the desired controller functionality.*

Filter

- Filters the controller output and computes the time derivative.
- *User modification is required for the implementation of the desired filter functionality.*

Output

- Compute the output signal that is provided to Cobalt. Depends on the control method (blowing/suction, displacement, plasma).
- *User modification is required for the implementation of the desired functionality.*

3.Ta p Extracts

Tap Locations (with extracts)

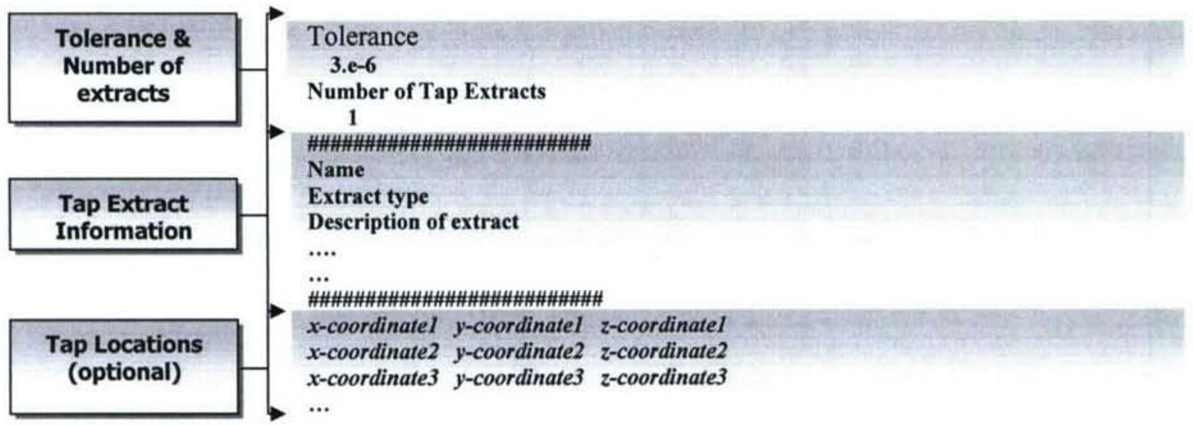
Tap extracts are a method to rapidly generate a set of taps within the domain using simple building blocks. They are specified using the same file as above, with additional information added. Specific tap locations are not requested – instead geometry shapes are given, and all points within that shape are output as tap locations. Using tap extracts does not preclude additionally specifying specific tap locations, as above. In this case, the tap extract information is inserted after the tolerance, and before the Tap locations. A sample tap file is provided: ***$CBLTLOC/reference/cobalt_extract.tap***.

The standard extension is .tap. An example is: **$CASEDIR/$CASE.tap **

Tap extracts are specified in the grid reference frame, and will remain fixed as the grid moves (i.e. the same points in the grid are output).

The backwards slash is required after a space at the end of the line.

Anatomy of a Tap File (with extracts): The contents of a tap file with extracts are show below.



```
Tolerance
  3.e-6
Number of Tap Extracts
    1
###########################
Name
Extract type
Description of extract
....
...
############################
x-coordinate1  y-coordinate1  z-coordinate1
x-coordinate2  y-coordinate2  z-coordinate2
x-coordinate3  y-coordinate3  z-coordinate3
...
```

Tolerance: This region specifies a "Tolerance" used to determine if a tap falls within a cell (see previous description), for the specified tap locations. This value is not used for the tap extracts, but must be specified.

Number of Tap Extracts: These keywords must be used to enable tap extracts. The following line then defines the number of tap extracts to be listed.

Name: This is a user-provided name used to describe the extract. It has no use other than to be echoed in the tap output, if HDF5 is selected as the output type.

Extract Type: The specific type of extract to be applied – see the following descriptions.

Tap Locations: Optional when using tap extracts, this region contains *n+1* lines where *n* is the number of taps. The first line is a header. The following *n* lines provide the coordinates of the taps. For a 3-D grid, the x-, y- and z-coordinates are required. For 2-D, only the x- and y-coordinates are required.

Cell

This extract specifies a region of the grid to be extracted that corresponds to a "super-cell". I.e. it is a cell described by a list of points, and all grid points within that cell will be output as taps.

Use the keyword: **Cell** to specify this method.

Sample Cell Extract Specification

A sample cell extract entry is shown below.

```
######################
Patch1
Cell
Number of Points (3D: 4,5,6,8, 2D: 3-4)
 5
x,y,z - See Documentation for point ordering
0.0 0.0 0.0
0.9 0.0 0.0
0.9 0.9 0.0
0.0 0.9 0.0
0.9 0.0 0.9
#######################
```

Number of Points

This entry specifies the number of points in the "super-cell". For 3-D, valid options are 4 (Tetrahedron), 5 (Pyramid), 6 (Prism), or 8 (Hexahedron). For 2-D, the valid options are 3 (Triangle), and 4 (Quadrilateral).

Coordinate list

The coordinates must be listed in a specific order.

Tetrahedron (4) – Order doesn't matter

Pyramid (5) – Four points on the bottom first, in a loop (clockwise or counter-clockwise). The fifth point is the tip of the pyramid.

Prism (6) – First four points as for the pyramid - the points on the four sided face. Point 5 should be the remaining point that shares an edge with point 1, Point 6 is the last point.

Hexahedron (8) – First four points trace a loop around one face. Then second four points trace a loop around the opposite face, with point 5 sharing an edge with point 1, 6 with 2, 7 with 3, 8 with 4.

Triangle (3) – Order does not matter.

Quadrilateral (4) – Order the points in a loop – direction does not matter (clockwise or counter clockwise).

Cone

This extract specifies a region of the grid to be extracted that corresponds to a cone. All gridpoints within that cone will be output as taps. Each end of the cone can have a different radius, so that cases such as a cylinder can be handled as well.

Use the keyword: **Cone** to specify this method.

Sample Cone Extract Specification

A sample cell extract entry is shown below.

```
########################
Patch2
Cone
 Center Point 1      Center Point 2     Radius 1      Radius 2
     0.0 0 0              2.0 0 0          2.0          4.0
#######################
```

Center Point 1 and 2

These entries specify the center points of each end of the cone.

Radius 1 and 2

These entries specify the radius around points 1 and 2 respectively. The cone axis lies along the vector from point 1 to 2, and the ends of the cone intersect points 1 and 2, and are perpendicular to the cone axis.

Surface Patches

This extract specifies a list of surface patches (or boundary condition patches) that will be output as tap extracts. All points on the specified boundary condition surfaces will be output as taps.

Use the keywords: **Surface Patches** to specify this method.

Sample Surface Patch Extract Specification

A sample surface patch extract entry is shown below.

```
#######################
Patch3
Surface Patches
List of Surface Patches
 4 11 12 13 14 15 16 17 18
########################
```

List of Patches

These entries specify the boundary condition patches that will be output as taps

4.St and Alone Codes

Stand alone codes are installed in $CBLTLOC/flowcontrol/bin

Parallel Proper Orthogonal Decomposition (ppod)

For a description of the capabilities of the Parallel POD code, please refer to the final STTR report in $CBLTLOC/flowcontrol/docs/sttr_report.pdf. The code will be executed through a job script – a sample is found in $CBLTLOC/flowcontol/examples/pod.job. The format of the job file is as follows:

A section that can be customized for your particular system/queing system.

```
#PBS -l nodes=1:a
#PBS -e stderr.e
#PBS -o stdout.o
cd ~/tutorial/flowcontrol
```

The next section sets the input and output files, and the variable to do POD on. The structure of the mtap file has a group for each tap extract, so the group name (based on the name we gave the extract in the tap file) must be included.

```
#INPUT DECK
cat > stdin << EOF
###  HDF Input File Name  ###
cylinder_runE.mtap
### Text Output File Name ###
cylinder_pod.out
###  HDF Output File Name ###
cylinder_pod.h5
###  Variable for POD ###
Patch1/U-velocity
```

The next section sets the parameters for the POD calculation. You may subtract the mean prior to performing the POD. For the case of "Normal" it will perform a POD on all the data. For SPOD it will use the indices listed. For DPOD, it will first perform an SPOD, then a DPOD on those SPOD modes. Finally, you may skip every N indices in order to calculate the POD more rapidly for datasets with large time series.

```
### Subtract Mean (0=No, 1=Yes) ###
0
### POD Type (0=Normal, 1=SPOD, 2=DPOD) ###
2
### Number of Modes / DPOD Modes ###
5   3
### DPOD Normalization Type (0=None, 1=by type, 2=by energy) ###
2
### Index Skip ###
1
```

Finally, indices are specified to do the different SPOD bins. Then manual ordering of set index pairs can be specified. Just give the index pair number, then list the unsorted modes in the order you want them sorted into. A negative number will multiply the mode by -1.

```
### Short Term POD Mode Indices ###
            84          114
           114         146
           146         176
           176         203
           203         229
...
          3491        3516
          3516        3542
        END
## Reference Frames and order ###
80 1 3 2 4 5
EOF
```

The next lines actually run the code, then clean up afterwards.

```
#RUN IT
mpirun -np 2 $CBLTLOC/flowcontrol/bin/ppod

#CLEAN UP
rm stdin
```

HDF to Fieldview converter

The code hdf2fv can convert appropriate hdf5 files into fieldview flow viz (results only) format. A fv grid file from the original run is required to view the results. This code can be used on a tap extract mtap file (in hdf5 format), or on the output of ppod. The arguments to hdf2fv are:

```
hdf2fv (-s) file.h5 fvbase
 -s - force write of SPOD even if DPOD exists
```

The first file is the input file, while the second name is a base file name that will have iteration number, the SPOD bin number, or the DPOD mode appended, as appropriate. Normally for POD modes, it will output the DPOD modes if they exist. This behaviour can be over-ridden by the –s flag which will force the code to convert the SPOD modes instead.

A variable "Blank" will be output in the fieldview files which is 0 wherever points were not part of the tap extract, and 1 where they were. This can be used to subset the visualization to restrict it to the extracted portions.

5.Tutori als

Tutorial #7: Flow Control of a Circular Cylinder

This tutorial is designed walk you through the use of the basic elements of the flow control toolbox. It is not comprehensive, but will cover the basic mechanics. It will take you through the entire process including POD model development, sensor placement, mode estimation, and feedback control.

This tutorial requires that you have done tutorials # 1 (steady cylinder), #2 (unsteady cylinder), and #5 (Pitching NACA 0012) from the main *Cobalt* manual. This will ensure you have the necessary experience to complete this tutorial. Additionally, files will be required from the second tutorial.

The case treated by this tutorial is based on the case considered in:

Siegel, Stefan; Cohen, Kelly; Seidel, Jürgen; Luchtenburg, Mark; McLaughlin, Thomas, "Low Dimensional Modeling of a Transient Cylinder Wake Using Double Proper Orthogonal Decomposition", accepted for publication in J. Fluid Mechanics, June 2007

This reference is available in $CBLTLOC/flowcontrol/docs.

For this case, the shedding of vortices on a cylinder at a Reynolds number of 100 (same as tutorial #2) will be reduced through feedback control by forced vertical motion. A model will be developed through the use of Double Proper Orthogonal Decomposition (DPOD), as described in the above reference. In order to get a robust model, the model will be developed through the use of the transient startup for the unforced case, as well as the transient response to forcing at different frequencies and amplitudes.

The various forcing cases considered will be the first five from the JFM article:

Case	Forcing Frequency/Shedding Frequency	Forcing Amplitude/Diameter
A	1	0.25
B	1	0.3
C	1.05	0.3
D	0.9	0.3
E	1.1	0.3

Table 1: Open Loop Cases

Setting up the Unforced Flow Job

The first case considered will be the unforced flow. The job file from Tutorial #2 will be the basis for the starting job. However, the timestep will be changed to $0.05*D/U_\infty$. This is an accurate, yet more aggressive timestep than what was used in the second tutorial. Based on the freestream velocity of 34.7 m/s and diameter of 1m, this results in a timestep of 1.4E-3 sec.

1. mkdir ~/tutorial/flowcontrol
2. cd ~/tutorial/flowcontrol
3. cp ../cylinder/cylinder_100.job cylinder_unf.job
4. Edit this new job file
5. Change the CASE to cylinder_unf
6. Make sure the initial Run flag is set to 1
7. Change the number of timesteps to 2000
8. Change timestep to 1.4E-3
9. Switch the time to start time averages to -1, since you won't be taking time-averages
10. Make sure the flow viz file type is set to fieldview since we will later want to convert the POD modes to fieldview format for viewing. Change the frequency of flow-viz files as desired.
11. Change the Provide Tap Data flag to 3 (for HDF5)
12. Change the list of tap data to 2 (U-velocity)
13. Change the Frequency of tap file output to 5 (every 5 iterations)
14. In the list of files, set the tap file to $CASEDIR/cylinder.tap

```
-----------------------------------------------------
                    TAP DATA AND FILE
-----------------------------------------------------
 PROVIDE TAP DATA?    LIST OF TAP DATA
    3                       2
 DATA IN LAB REF FRAME?    DATA TIME-AVERAGED?
     0                          0
 FREQUENCY (IN TIME-STEPS) OF TAP FILE OUTPUT
    5
```

Note that if you put in the full path to the grid and bc files, you should not need to copy this into the new directory.

Setting up the Tap Extract File

In order to create the POD model, we will set up a tap file to extract data in the wake of the cylinder. To do this "Tap Extracts" will be used to simplify the process (see the User's manual on Tap file formats). Create the file cylinder.tap, and input the following:

```
Tolerance
   3.e-6
Number of Tap Extracts
   1
```

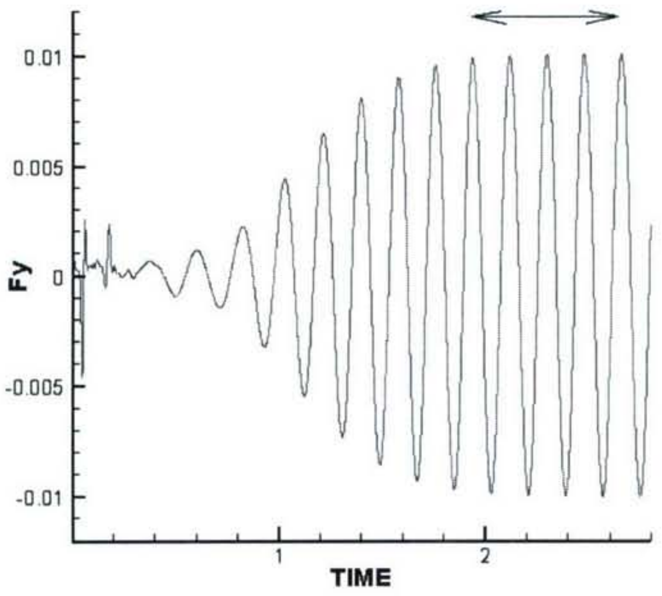
```
######################
Patch1
Cell
Number of Points (3D: 4,5,6,8, 2D: 3-4)
4
x,y,z - See Documentation for point ordering
-1.0 -2.0
-1.0 2.0
6.5 2.0
6.5 -2.0
#########################
```

This will create a rectangular region behind the cylinder for extracting the data.

Run the Job

Run the job by submitting it to the scheduler, or running it on the command line. After the job has run, open the cylinder_unf.cnvrg/total.cnvrg file in your favorite plotting program, and plot the y-forces (Fy) vs. time. Measure a few cycles, and divide the time elapsed by the number of cycles to determine the shedding period. From the period, the frequency can be determined (1/T). This frequency will then be used to set up the open loop forced runs.

T=0.1775s
F=5.64 Hz



Setting up the Motion File

For the subsequent open loop forced cases, a motion file for each must be set up. For each case, you will turn the controller on at time=0 through the use of a shifted cosing, and leave it on for 15 forcing cycles, then off for an additional 10.

This will expose the flow to a startup and stopping transient. The fragment of the motion file that needs to be calculated based on Table 1 is:

```
Time          Current Orientation (X^,Y^,Z^ Axes)                    Center of Rotation
```

The current orientation is the transformation matrix from the original position, thus it as 3x3 matrix, that should start as the identity matrix. In our case, since we aren't doing rotation, it can be left as the identity matrix. The center of Rotation is also used to translate the grid. Therefore it should start out as 0 0 0, and then the y location should be varied as a shifted cosine. The following matlab routine will create a mtn.dat file that contains the entries needed to fill in that section of the motion file. The first two lines set the frequency and amplitude and should be picked according to Table 1.

```
f=5.64*1.0;
A=0.25;
T=1/f;
Delta_T=1.4E-3;
Ncycle_on=15;
Ncycle_off=10;
N_per_cycle=80
clear Y Time data;
data=zeros(N_per_cycle*Ncycle_on+1,13);
data(:,2)=1.0;
data(:,6)=1.0;
data(:,10)=1.0;
for i=1:N_per_cycle*Ncycle_on+1
    Time(i)=T*(i-1)/N_per_cycle;
    Y(i)=A*(cos(2*pi*(i-1)/N_per_cycle)-1);
    data(i,1)=Time(i);
    data(i,12)=Y(i);
end
plot(Time,Y);
endmtn=Ncycle_on*T
Niter=T*(Ncycle_on+Ncycle_off)/Delta_T
save mtn.dat data -ASCII
```

This will also tell you the # of iterations required for the run (Niter), which will be used in the job file.

After running this code:

1. cp $CBLTLOC/examples/cobalt.mtn cylinder_runA.mtn
2. Edit cylinder_runA.mtn and remove all other motion types, other than the arbitrary motion
3. Copy the data from mtn.dat into the appropriate section in the motion file.
4. copy the final time from mtn.dat into the "Stop Time" for the motion
5. Set the start time to 0.0

The file should look as follows.

```
##################################################
   Motion Specification File for:
   Cobalt version 3.0
##################################################
  Force & Moment Axes
   X' Axis
    1.0  0.0  0.0
   Y' Axis
    0.0  1.0  0.0
   Z' Axis
    0.0  0.0  1.0
  Are Force & Moment Axes Subject to Motions (Yes or No)?
    Yes
##################################################
  User-Defined Axis System
   X" Axis
    1.0  0.0  0.0
   Y" Axis
    0.0  1.0  0.0
   Z" Axis
    0.0  0.0  1.0
  Origin of User-Defined Axis System
    0.0  0.0  0.0
##################################################
  Mass   Center of Mass (in Lab Ref Frame @ t = 0)
  0.0000        0.0  0.0  0.0
##################################################
  Motion Type
   Arbitrary
  Start Time    End Time
    0.0          2.6595745e+000
   Time        Current Orientation (X^,Y^,Z^ Axes)                 Center of Rotation
  0.0000000e+000  1.0000000e+000  0.0000000e+000  0.0000000e+000  0.0000000e+000 …
  2.2163121e-003  1.0000000e+000  0.0000000e+000  0.0000000e+000  0.0000000e+000 …
  4.4326241e-003  1.0000000e+000  0.0000000e+000  0.0000000e+000  0.0000000e+000 …
  6.6489362e-003  1.0000000e+000  0.0000000e+000  0.0000000e+000  0.0000000e+000 …
...
  2.6595745e+000  1.0000000e+000  0.0000000e+000  0.0000000e+000  0.0000000e+000 …
 End
##################################################
```

Setting up the Open Loop Job file

For the open loop jobs, we will run each one sequentially in order to collect one single time series for POD create. For each case we will therefore copy the restart and mtap files from the previous runs. But for each run, the time will be reset to zero, so that the motion start times can always simply be 0.0. To create the first job (runA), therefore:

1. Change the CASE name from cylinder_unf to cylinder_runA
2. Change the Start Option to 3 (start from a restart, set time to 0)
3. Change the number of iterations to the output of Niter determined when creating the motion file
4. Change the number of Newton Subiterations to 5 since we will be doing motion
5. Note that the mtn file is set to $CASE.mtn, so this does not need to be changed.

Afterwards copy the restart and mtap files over:

1. cp cylinder_unf.rst cylinder_runA.rst
2. cp cylinder_unf.mtap cylinder_runA.mtap

Then run the job. For subsequent jobs (B-E), you will make the motion file as described above based on Table 1. Then you can start from the previous job. E.g.

1. cp cylinder_runA.job cylinder_runB.job
2. Change the CASE name from cylinder_runA to cylinder_runB
3. Change the number of iterations to the output of Niter determined when creating the motion file
4. cp cylinder_runA.rst cylinder_runB.rst
5. cp cylinder_runA.mtap cylinder_runB.mtap

Run each job in sequence.

POD Calculation

After the jobs have run, you will need to segment the data in order to perform short term POD (SPOD). Do this by first reading in the convergence files for each run into Matlab:

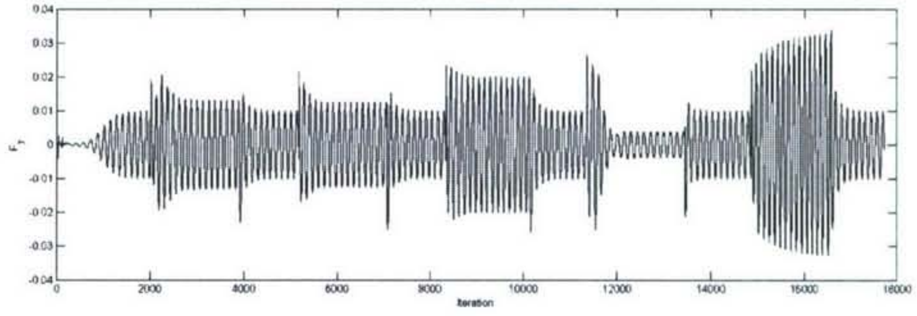
1. Select File-import in matlab
2. Open the total.cnvrg file in the .cnvrg directory
3. Select space delimited, and specify two header lines
4. Select next, - rename data to unf, runA, runB, etc as appropriate
5. Click finish to bring in the data

We will then be using the supplied Matlab routine PEAKDET to detect peaks in order to segment the run:

```
PEAKDET Detect peaks in a vector
         [MAXTAB, MINTAB] = PEAKDET(V, DELTA) finds the local
         maxima and minima ("peaks") in the vector V.
         A point is considered a maximum peak if it has the maximal
         value, and was preceded (to the left) by a value lower by
         DELTA. MAXTAB and MINTAB consists of two columns. Column 1
         contains indices in V, and column 2 the found values.
```

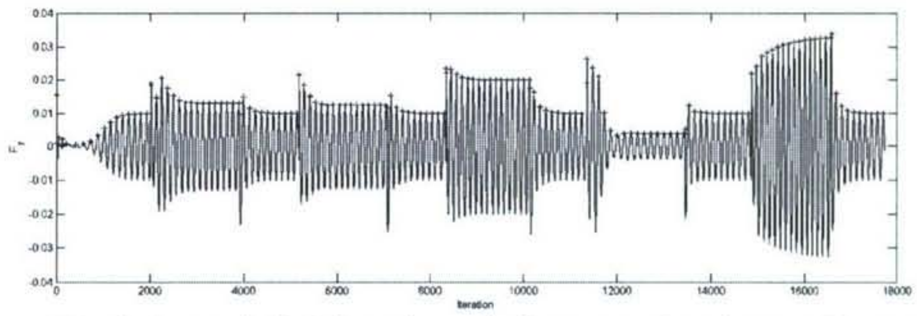
First, look at the headers of any total.cnvrg file to recognize which variables are which. You can then create a variable for the time histories of the normal force, Iteration #, and y-translation by:

```
>> Fy=[unf(:,8);runA(:,8);runB(:,8);runC(:,8);runD(:,8);runE(:,8)];
>> Oy=[unf(:,23);runA(:,23);runB(:,23);runC(:,23);runD(:,23);runE(:,23)];
>> Iter=[unf(:,1);runA(:,1);runB(:,1);runC(:,1);runD(:,1);runE(:,1)];
>> plot(Fy)
```



Based on the plot above, a DELTA of 0.001 should be small enough to capture the early peaks, yet not too small to capture small wiggles in the data. Detect the peaks, then plot the maximum points against the original data.

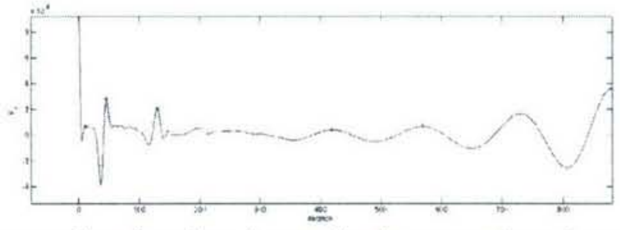
```
>> [maxtab mintab]=peakdet(Fy,0.001);
>> hold on
>> plot(maxtab(:,1),maxtab(:,2),'+')
```



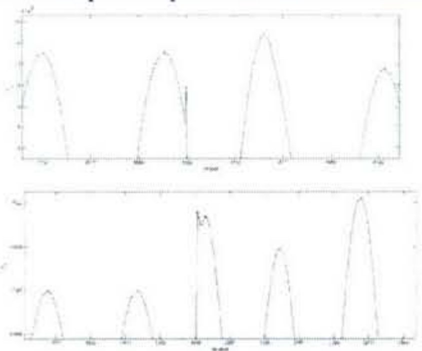
What is then needed is index pairs – pairs from one peak to the next. These can be created by running the following matlab code:

```
for j=1:size(maxtab,1)-1
indices(j,1)=maxtab(j,1);
indices(j,2)=maxtab(j+1,1);
end
```

Sometimes multiple neighboring peaks are detected due to small wiggles in the data, or the transition from one run to the next. These should be cleaned up prior to using them for the POD calculation. To do this, open indices with the array editor and keep the figure displayed. Zoom in to the plot, and look for repeated indices, then delete the relevant index pairs in the array editor. This will leave small gaps where there are these small spikes. At the beginning, the indices prior to the start of the oscillatory motion should be removed.



Some examples of portions that need to be removed are shown below.



These transients occur because of the starting or stopping of the motion abruptly. Once the array indices is cleaned up, it needs to be renormalized since only every 5th iteration was output to the hdf tap file.

```
>>indices=round(indices/5)
```

Next, the PPOD job needs to be created, using these indices to determine the SPOD bins. The job will use as input the mtap file from the final run – i.e. cylinder_runE.mtap. Copy the sample pod job over

1. cp $CBLTLOC/flowcontrol/examples/pod.job cylinder_pod.job

Edit the file and modify the top section as appropriate for your scheduling system:

```
#PBS -l nodes=1:a
#PBS -e stderr.e
#PBS -o stdout.o
cd ~/tutorial/flowcontrol
```

Then set the input and output files, and the variable to do POD on. The structure of the mtap file has a group for each tap extract, so the group name (based on the name we gave the extract in the tap file) must be included.

```
#INPUT DECK
cat > stdin << EOF
###  HDF Input File Name  ###
cylinder_runE.mtap
### Text Output File Name ###
cylinder_pod.out
###  HDF Output File Name ###
cylinder_pod.h5
###  Variable for POD ###
Patch1/U-velocity
```

Next the parameters for the POD calculation are set. We do not want to subtract the mean from the results since we will be doing DPOD. The POD type will be DPOD. For the case of "Normal" it will perform a POD on all the data. For SPOD it will use the indices listed. For DPOD, it will first perform an SPOD, then a DPOD on those SPOD modes. We will use 5 Modes for the POD, and 3 modes for the DPOD (the main and two shift modes). Select normalization by energy. Finally, PPOD allows you to skip every N indices in order to calculate the POD more rapidly for datasets with large time series. In this case, we want every frame to be used (we already output only ever 5 iterations).

```
### Subtract Mean (0=No, 1=Yes) ###
0
### POD Type (0=Normal, 1=SPOD, 2=DPOD) ###
2
### Number of Modes / DPOD Modes ###
5   3
### DPOD Normalization Type (0=None, 1=by type, 2=by energy) ###
2
### Index Skip ###
1
```

Finally, copy the indices from matlab and paste them into the SPOD mode indices section. Set the correct # of processors. Leave the Reference Frames and order section blank for now.

```
### Short Term POD Mode Indices ###
          84          114
         114          146
         146          176
         176          203
         203          229
...
        3491         3516
        3516         3542
      END
## Reference Frames and order ###
EOF

#RUN IT
mpirun -np 2 $CBLTLOC/flowcontrol/bin/ppod

#CLEAN UP
rm stdin
```

Submit this job to your scheduler, or run it command line as appropriate. While the code is running it will dump output into cylinder_pod.out. You can watch this file as the code progresses. You may get errors that say you don't have enough frames (you need at least as many frames as the # of modes you are requesting):

```
  INVALID INDEX PAIR (NOT ENOUGH SNAPSHOTS):        53
  INVALID INDEX PAIR (NOT ENOUGH SNAPSHOTS):        80
  INVALID INDEX PAIR (NOT ENOUGH SNAPSHOTS):        91
  INVALID INDEX PAIR (NOT ENOUGH SNAPSHOTS):       135
```

If this happens, then remove the offending index pairs from the job file, and rerun.

Inspect the output file, and you will see that the SPOD modes are created, then sorted. The sorting is done by doing a spatial correlation of modes in one bin with the modes in the next bin. They are rearranged to give the strongest correlation. If a negative correlation occurs, the sign is flipped. In some cases, one mode may be the best correlation with two other modes, in which case you will get a warning that the mode has already been taken. The modes are sorted in reverse order, since the final flow state may be more orderly than the initial transient startup.

```
  Swapping Modes         4 and    5 for SPOD     117
  Flipping Mode          4 for SPOD    117
  Flipping Mode          5 for SPOD    117
  !!! Warning, Mode      5 best match already taken for SPOD    118
  Flipping Mode          2 for SPOD    116
  Flipping Mode          3 for SPOD    116
```

The final correlations after sorting are reported.

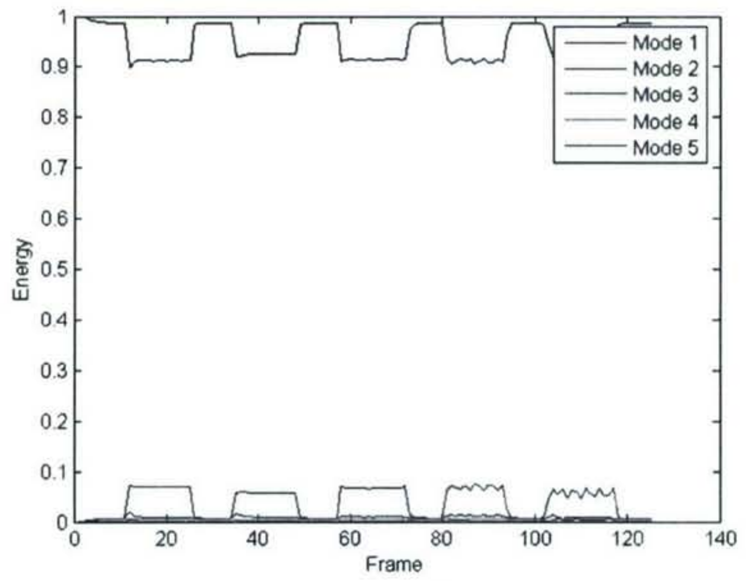
```
### FINAL CORRELATION INFORMATION ###
  Correlation of current mode with same mode in next SPOD bin
  SPOD BINS|  MODES - >
          V
         125 1.000 0.999 0.999 0.972 0.972
         124 1.000 1.000 1.000 0.973 0.974
         123 1.000 1.000 0.999 0.976 0.976
         122 1.000 1.000 1.000 0.974 0.974
         121 1.000 0.998 0.998 0.987 0.988
         120 1.000 0.995 0.995 0.950 0.951
         119 0.999 0.973 0.975 0.950 0.957
         118 0.981 0.686 0.674 0.371 0.203
         117 0.993 0.992 0.900 0.902 0.885
         ...
```

Unfortunately the sorting process is not fool-proof. During transient portions of the simulation, the correlations may be very weak, and modes can get fooled into following the wrong mode. This will show up in plots of energy, since normally the lowest numbered modes should have the highest energy. This may not be true during a transient (therefore sorting based strictly on energy won't work either), but should be true on average. To plot the energy take a look at the structure of the output POD file:

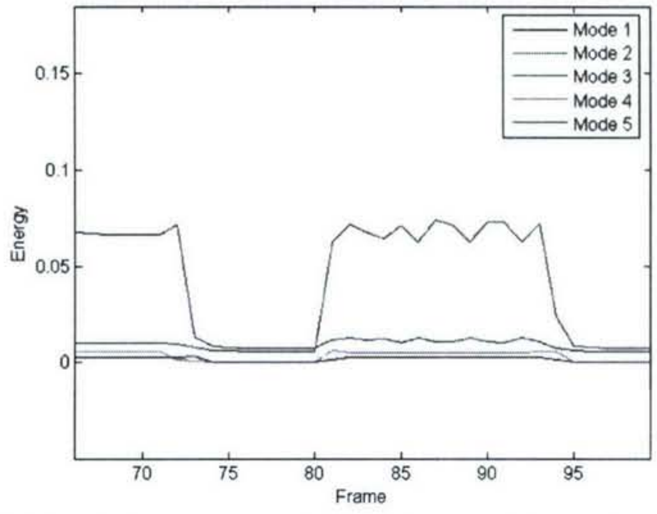
```
>>hdf5struct('cylinder_pod.h5')

ans =

  cylinder_pod.h5
    /
      /Patch1-U-velocity
        /Patch1-U-velocity/Correlation
         Dims 5    5  125
        /Patch1-U-velocity/DPOD_Energy
         Dims 6  5
        /Patch1-U-velocity/DPOD_Modes
         Dims 6      5  21455
        /Patch1-U-velocity/Energy
         Dims 125    5
    /Patch1-U-velocity/Global Indi
     Dims 21455
    /Patch1-U-velocity/Modes
     Dims 125       5  21455
  /Total number of points
  /Number of dimensions
```

Then read the energy of the SPOD modes (the ones not labeled as DPOD), and plot them:

```
>> Energy=hdf5read('cylinder_pod.h5','/Patch1-U-velocity/Energy');
>>plot(Energy);xlabel('Frame');ylabel('Energy');legend('Mode 1','Mode
2','Mode 3','Mode 4','Mode 5')
```



Note that Mode 1, the mean mode, has by far the most energy, as expected. Looking at Mode 2, for the last two runs, it is in the appropriate order, but prior to that it has swapped. Zooming in on the transient region around frame 80



So it looks like mode 2 and 3 need to be swapped around frame 80. These results may be slightly different based on the indices you retained. So at frame 80, we want to force PPOD to make the third mode go to the second, and vice versa. Do this by adding near the bottom of the job file:

```
## Reference Frames and order ###
80 1 3 2 4 5
EOF
```

You should also inspect modes 4 and 5 to ensure they retain the proper order throughout the run. Remember that modes are sorted starting at the ending bin. After rerunning the pod job, you should see an entry during the sort:

```
  Manually sorting SPOD      80
```

Check the correlation in the output file to make sure that mode 80 is positively correlated. If not you can specify a flipping of a mode by putting in a negative sign, e.g.

```
## Reference Frames and order ###
80 1 -3 2 4 5
EOF
```

Plotting the SPOD/DPOD Modes with hdf2fv

The next step is to plot the modes to check that they make sense. This can be done by using the utilty hdf2fv which will convert the hdf file into a series of fieldview files. The arguments to hdf2fv are:

```
hdf2fv (-s) file.h5 fvbase
 -s - force write of SPOD even if DPOD exists
```

So first create the SPOD modes:

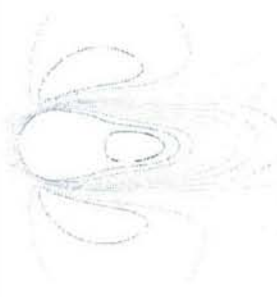
```
hdf2fv  -s cylinder_pod.h5 cylinder_spod
```

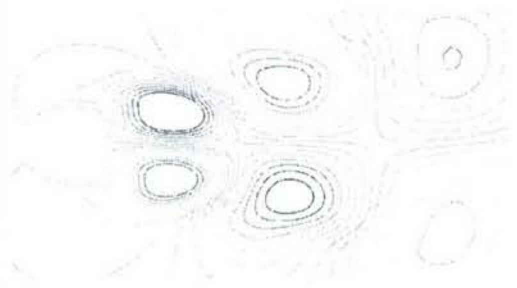
```
  Fieldview to HDF, Version: 7/27/07
 HDF file name: cylinder_pod.h5
 FV file base: cylinder_spod
 Number of Dimensions:              2
 Total Number of Points:        31948
 Extracts
  Patch1-U-velocity
 Flow Variables
  Patch1-U-velocity_mode_1
  Patch1-U-velocity_mode_2
  Patch1-U-velocity_mode_3
  Patch1-U-velocity_mode_4
  Patch1-U-velocity_mode_5
  Blank
 Writing cylinder_spod1.uns
 Writing cylinder_spod2.uns
 Writing cylinder_spod3.uns
...
```

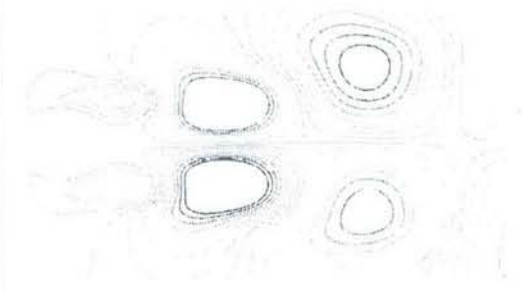
Then you can make the DPOD modes by omitting the –s flag and changing the fv base name.

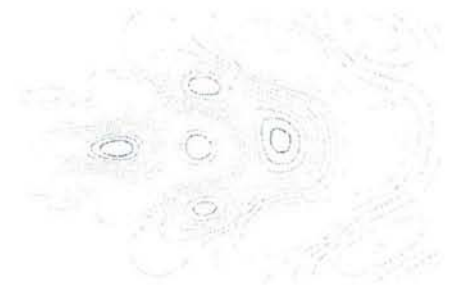
```
hdf2fv cylinder_pod.h5 cylinder_dpod
  Fieldview to HDF, Version: 7/27/07
 HDF file name: cylinder_pod.h5
 FV file base: cylinder_dpod
 Number of Dimensions:              2
 Total Number of Points:        31948
 Extracts
  Patch1-U-velocity
 Flow Variables
  Patch1-U-velocity_dpod_1
  Patch1-U-velocity_dpod_2
  Patch1-U-velocity_dpod_3
  Patch1-U-velocity_dpod_4
  Patch1-U-velocity_dpod_5
  Blank
 Writing cylinder_dpod1.uns
 Writing cylinder_dpod2.uns
 Writing cylinder_dpod3.uns
```

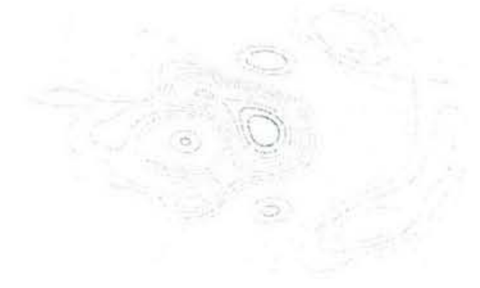
These files can then be viewed in fieldview by reading the grid file in from any one of the runs, then the POD modes as the results files, which will be read as a transient set. The bin number (in the case of the POD) or the DPOD mode number are the transient index that can be used to create animations. Below is a plot of all 5 modes for a single SPOD bin (single fv file).


Mode 1

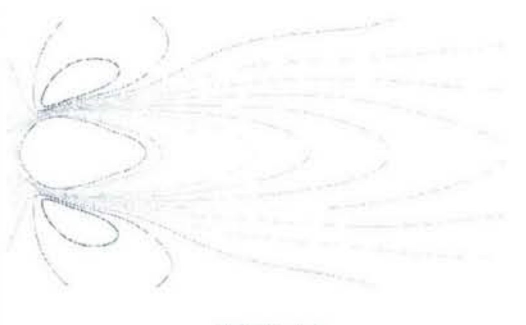

Mode 2

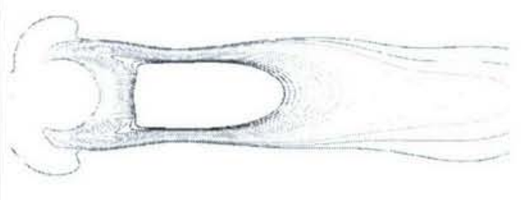

Mode 3

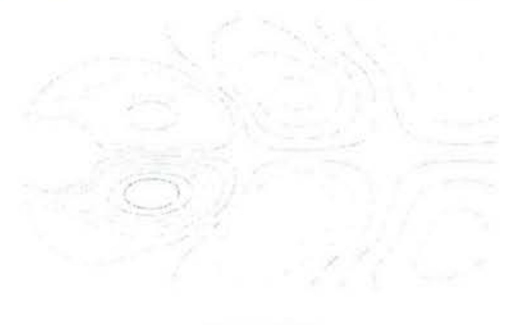

Mode 4

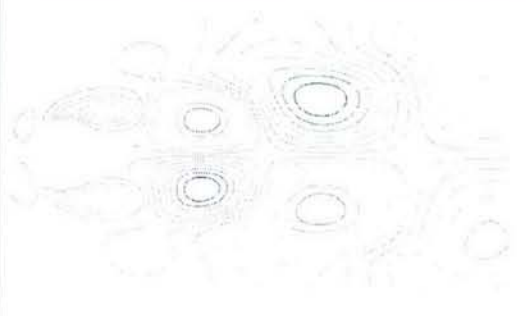

Mode 5

Plotting the DPOD Modes (i.e cylinder_dpod*.uns) results in the following first two modes, and their shift modes:


Mode 1 1


Mode 1 2 (shift mode)


Mode 2 1


Mode 2 2 (shift mode)

If any problems are observed with the sorting of the modes, some more may need to manually sorted, as described previously.

Setting up the Control Case – Mode File

In order to perform closed loop control, a Matlab file must be created the contains the Modes, the coordinates of the points for the modes, and the mean flow. In our case, we have not subtracted out the mean, so it can simply be set to zero. First we will extract the modes. Examine the contents of the POD hdf file:

```
>> hdf5struct ('cylinder_pod.h5')

ans =

  cylinder_pod.h5
    /
      /Patch1-U-velocity
        /Patch1-U-velocity/Correlation
         Dims 5    3  125
        /Patch1-U-velocity/DPOD_Energy
         Dims 6  3
        /Patch1-U-velocity/DPOD_Modes
         Dims 6      3  21455
        /Patch1-U-velocity/Energy
         Dims 125    3
      /Patch1-U-velocity/Global Indi
       Dims 21455
      /Patch1-U-velocity/Modes
       Dims 125      5  21455
    /Total number of points
    /Number of dimensions
```

From this we can see where the DPOD modes are and read them. Hdf5read does not preserve the index ordering however, so the array needs to be permuted to get into the proper order (# points, # Modes, # DPOD Modes). Then the modes need to be collapsed into one single set of modes (so Modes 1-5 are modes 1 1, 2 1, 3 1, etc). This will lead to 15 modes (5x3).

```
>>Modes=hdf5read('cylinder_pod.h5','/Patch1-U-velocity/DPOD_Modes');
>>size(Modes)
ans =

           5       21455            3
>>Modes=permute(Modes,[2 1 3]);
>>Modes=reshape(Modes,21455,15);
```

Then the XYZ's need to be read. For all the moving grid cases, there will be a separate X,Y,Z for each iteration, moving with the grid. However, we want just the original position, which we can do by reading from the mtap file for the unforced case:

```
>> hdf5struct('cylinder_unf.mtap')

ans =

  cylinder_unf.mtap
    /
      /Patch1
        /Patch1/Global Indices
         Dims 21455
        /Patch1/Tap Coordinates
         Dims 1  21455      3
        /Patch1/U-velocity
         Dims 400  21455
      /Iteration
       Dims 400
      /Time
       Dims 400
      /Number of dimensions
      /Total number of points
      /Checksum

>> XYZ=hdf5read('cylinder_unf.mtap','/Patch1/Tap Coordinates');
```

Finally, the Mean mode can be create as a matrix of the appropriate size (# of points) filled with zeros. All arrays are then redeclared as doubles, and saved in a matlab file.

```
>>Modes=double(Modes);
>>XYZ=double(XYZ);
>>Mean=zeros(size(Modes,1),1);
>>save CYL_DPOD_TUT.mat Modes Mean XYZ
```

The Matlab file is then copied to the standard location.

```
cp CYL_DPOD_TUT.mat ~/FlowControl/Fbk/Modes
```

Setting up the Control Case – Motion File

For the motion file, start by copying one of the previous motion files.

1. cp cylinder_runE.mtn cylinder_feed.mtn

Edit the file, and remove the arbitrary motion section, and insert the following in its place. This will set up an array of sensors that has a Cartesian grid layout from x=0.75 to 6.75 and y=-1.0 to 1.0, in a 7x5 grid respectively. The Feedback start time is zero, and the end time set to something large.

```
#################################################
  Motion Type
   Feedback-Controlled
  Start Time     End Time
   0.0            100.0
  Sensors fixed to Lab or Body Reference Frame?
   Lab
  Sensor Locations
  7.5000000e-001 -1.0000000e+000  0.0000000e+000
  7.5000000e-001 -5.0000000e-001  0.0000000e+000
  7.5000000e-001  0.0000000e+000  0.0000000e+000
  7.5000000e-001  5.0000000e-001  0.0000000e+000
  7.5000000e-001  1.0000000e+000  0.0000000e+000
  1.7500000e+000 -1.0000000e+000  0.0000000e+000
  1.7500000e+000 -5.0000000e-001  0.0000000e+000
  1.7500000e+000  0.0000000e+000  0.0000000e+000
  1.7500000e+000  5.0000000e-001  0.0000000e+000
  1.7500000e+000  1.0000000e+000  0.0000000e+000
  2.7500000e+000 -1.0000000e+000  0.0000000e+000
  2.7500000e+000 -5.0000000e-001  0.0000000e+000
  2.7500000e+000  0.0000000e+000  0.0000000e+000
  2.7500000e+000  5.0000000e-001  0.0000000e+000
  2.7500000e+000  1.0000000e+000  0.0000000e+000
  3.7500000e+000 -1.0000000e+000  0.0000000e+000
  3.7500000e+000 -5.0000000e-001  0.0000000e+000
  3.7500000e+000  0.0000000e+000  0.0000000e+000
  3.7500000e+000  5.0000000e-001  0.0000000e+000
  3.7500000e+000  1.0000000e+000  0.0000000e+000
  4.7500000e+000 -1.0000000e+000  0.0000000e+000
  4.7500000e+000 -5.0000000e-001  0.0000000e+000
  4.7500000e+000  0.0000000e+000  0.0000000e+000
  4.7500000e+000  5.0000000e-001  0.0000000e+000
  4.7500000e+000  1.0000000e+000  0.0000000e+000
  5.7500000e+000 -1.0000000e+000  0.0000000e+000
  5.7500000e+000 -5.0000000e-001  0.0000000e+000
  5.7500000e+000  0.0000000e+000  0.0000000e+000
  5.7500000e+000  5.0000000e-001  0.0000000e+000
  5.7500000e+000  1.0000000e+000  0.0000000e+000
  6.7500000e+000 -1.0000000e+000  0.0000000e+000
  6.7500000e+000 -5.0000000e-001  0.0000000e+000
  6.7500000e+000  0.0000000e+000  0.0000000e+000
  6.7500000e+000  5.0000000e-001  0.0000000e+000
  6.7500000e+000  1.0000000e+000  0.0000000e+000
  End
#################################################
```

Setting up the Control Case – Job File

We will start the controlled case from the unforced case, and reset the time to zero. A matlab file will be used to set up the relevant parameters in Matlab. A simple proportional differential controller will be used based on the second mode (2 1). Different phases will be used to determine the optimum phase angle for the controller. We will therefore label each job with the phase angle, starting with phase=0.

1. cp cylinder_runA.job cylinder_feed0.job
2. change the CASE name to cylinder_feed0_ (the extra underscore is to prevent the iteration # in the fieldview file from getting appended to the 0)
3. Change the number of iterations to 2000
4. In the list of filenames at the bottom, change the motion file to $CASEDIR/cylinder_feed.mtn – since this file will not change for each case
5. Change the control file to point to $CASEDIR/$CASE.m

```
$CBLTLOC/CoMPIRUN \
linux \
double \
$CASEDIR/$CASE.inp \
$CASEDIR/cylinder.bc \
$CASEDIR/cylinder_unsteady.grd \
$CASEDIR/cylinder_feed.mtn \
$CASEDIR/$CASE.prf \
$CASEDIR/cylinder.tap \
$CASEDIR/$CASE.out \
$CASEDIR/$CASE.m \
$CASEDIR/$CASE.1dhe \
$CASEDIR/$CASE.mtap \
$CASEDIR/$CASE.rst \
$CASEDIR/$CASE.cnvrg \
$CASEDIR/viz
```

This job file can then be copied to cylinder_feed30, cylinder_feed_60, etc. for the different phase angles (from 0 to 330 in 30 degree increments). The only thing that needs to be change in each file then is the CASE name, to match.

Setting up the Control Case – Matlab Control File

Finally, the Matlab control file needs to be set up for each case ($CASE.m). Start with the sample file from $CBLTLOC/flowcontrol/control.m.

1. cp $CBLTLOC/flowcontrol/control.m. cylinder_feed.m

Edit this file, which is split into several sections. This file can be varied for whatever application you are interested in – it will be executed at the beginning of the CFD run by Matlab. First is the section of settings for any run, and is fairly self explanatory. The controller will turn on at t=0, and off at t=4.2. which should be after our run is completed.

```
% Settings for any Run
Run.MaxDYdt              = 0.025;     % Limits rate of change of position between time steps

Run.NTimesteps           = 5001;
Run.LogFrequency         = 1;         % log file  will be written every nth time step,
                                      % or only in the end if -1
Run.Display              = 0;         % disables the display

Run.TimeContOn           = 0.0;       % time when the feedback loop will be closed
Run.TimeContOff          = 4.2;       % time at which Feedback loop will be disabled
```

The next section sets up the modules that will be run. Run.ModeModule is the module that converts the sensor measurements into mode amplitudes. In this case it uses least squares fittings. The Control module is next – a variable gain, variable phase proportional differential controller. In our case, we will use a fixed gain and phase by setting the beginning and end gain and phase to be the same. The filter module then filters the controller output in order to reduce noise in the output signal. The final module (OutputModule) determines the controller output – y displacement in this case.

```
% the modules to be used for this run
Run.ModeModule           = 'MF_LSQ_UNS';
        % an M File that converts Sensor Readings to Mode Amplitudes and their derivatives.
        % Calling Convention:
        % [ModeAmp, ModeAmpDt] = MF_LSQ_I(SensorData, Dt, ModeAmpLog, ModeAmpDtLog, Mode)
Run.ContModule           = 'VarGainVarPhaseV3';
        % Name of the Control algorithm to be used
        % Calling Convention:
        % [ContOut, ContLog] = VarGainVarPhase(ModeAmp,ModeAmpDt,ContLog, Cont);
Run.FilterModule         = 'LowPass';
        % Filters the Controller Output and calculates d/dt, turns
        % Controller on and off
        % Calling Convention
        % [FiContOut, FiContOutDt, FiLog, FiDtLog] = LowPass(ContOut, FiLog, FiDtLog, Filter);
Run.OutputModule         = 'YDisp_V4';
        % an M file that uses filtered controller output and its time derivative to update the hdf
file.
        % Calling Convention example:
        % Status = YDisp(FiContOut, FiContOutDt, Out);
```

The final section has settings for the Modules. The Mode.VelComp is the component of the sensors that is used for feedback (which should match what was used to create the modes).

Density	4
U-Velocity	5
V-Velocity	6
W-Velocity	7
Pressure	8

The next section defines a time to start a ramp up of the controller output, and when it will reach the full value. For this particular controller, a hyperbolic secant function is used to ramp up smoothly with no abrupt start. This ramp up should take place over a few cycles of shedding in order to give the flow time to respond to the controller.

```
% Mode Filter Module Settings
 Mode.VelComp = 5;
 Mode.ModeFile = 'CYL_DPOD_TUT.mat';

% Controller Parameters
% VarGainVarPhase Controller
 Cont.TimeRampStart       = 0.0;       % time when the gain has value zero
 Cont.TimeRampEnd         = 0.5;       % time when the gain reaches its full value of *Gain
```

The next entries specify which mode will be used or feedback. In this case we take mode 2 (which is mode 2 1 of the DPOD modes). The shift mode of this mode would normally be used to vary the phase (mode 6, which is DPOD mode 1 2).

```
 Cont.FluctMode           = 2;         % Which mode to use for feedback
 Cont.SteadyMode          = 6;         % Which mode to use to adjust Phase
 Cont.SteadyModeOffset    = 0;
```

Next the gains and phases are set. We set the initial and final gains and phases equal since we will be doing constant phase/gain control.

```
 Cont.InitialGain        = 0.25e-3;      % Proportional Gain at beginning of Run
 Cont.AmplitudeGain      = 0.0          % Change in Gain per mode amplitude
 Cont.FinalGain          = 0.25e-3;      % limits the gain in/decrease

 Cont.InitialPhase       = 0;          % Phase advance at beginning of Run
 Cont.PhaseGain          = 0.0;        % Phase modification in deg per mode amplitude
 Cont.FinalPhase         = 0;          % limits the maximum amount of phase advance
```

Next set the natural frequency of the shedding, in order to scale the gains. There are then some settings that make plotting of the data after the fact easier. Then the Filter kernel is chosen, with a scaling value. There are then no variables required for the output module.

```
 Cont.Frequency           = 5.64;        % used to scale P and D gain ratio

 Cont.LegendStrg           = strvcat('Feedback Mode','d Mode/ dt /100','PGain *10000', 'DGain
*1000000', 'Phase Adv', 'MGain');
 Cont.Scaling              = [1 0.01 10000 1000000 1 10000];

% Filter Module Settings
  Filter.FilterFile = 'Fbk_PM_FilterKernel E112_0456.mat';
  Filter.FilterScaling     = 1;

% Output Module settings
%   YDisp V4 - no variables required
```

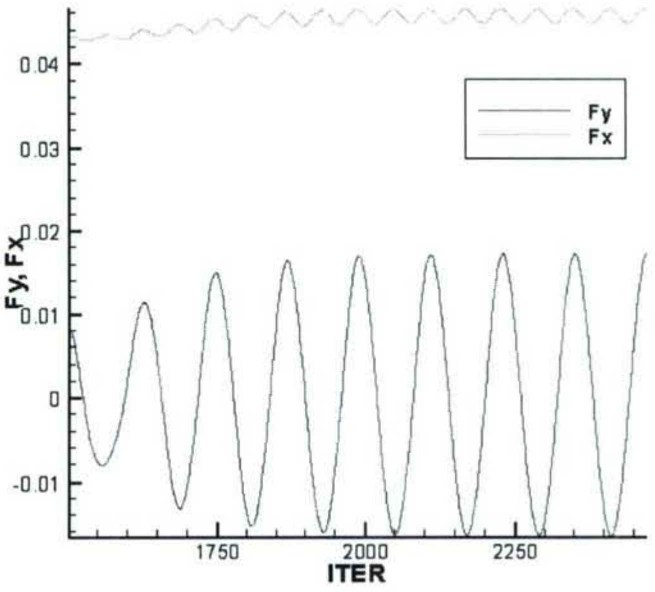
Once this matlab file is set up, you can copy it over to matlab_feed30.m, and then change the phase (as highlighted above) to 30. Repeat this for 60, 90, … 330.

Running the Control case

For each case, you will need to copy over the restart file and then submit the job. For example (assuming qsub is the means to submit a job – replace this as appropriate)

1. cp cylinder_unf.rst cylinder_feed0.rst
2. qsub cylinder_feed0.job

Let the jobs run, then go into the convergence directories of each and plot the convergence history for Fx and Fy. For example, for phase 0:

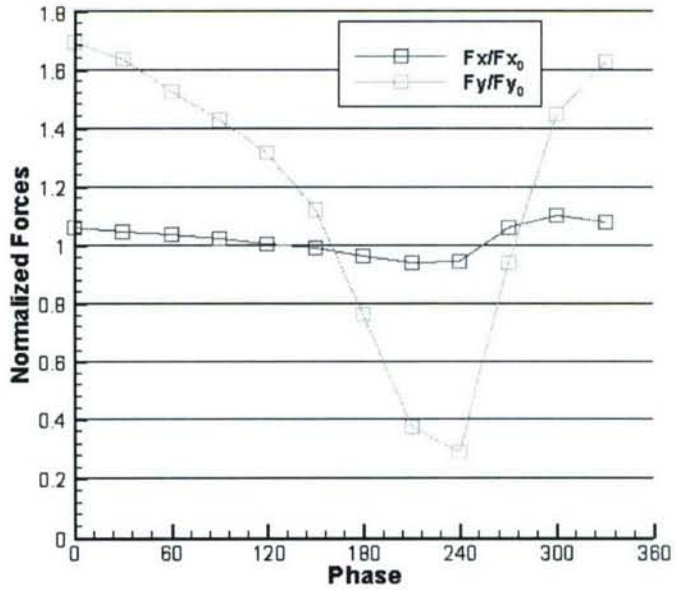


Note that in this case, the lift oscillations increase, as does the drag. Measure the peak amplitudes once it reaches stationary flow of the lift, and then the average of the drag. Do this for all cases, and for the unforced case. The results are in the table below. Your results may vary based on how you segmented the POD data, which can lead to slightly different POD modes, giving the phase a different meaning.

Phase	Peak Fy	Average Fx
unforced	0.0101	0.043
0	0.0171	0.0455
30	0.0165	0.0449
60	0.0154	0.0445
90	0.0144	0.0439
120	0.0133	0.0432
150	0.0113	0.0426

180	0.0077	0.0413
210	0.00376	0.0403
240	0.00295	0.0406
270	0.0095	0.0456
300	0.0146	0.0473
330	0.0164	0.0463

Take this data, and divide the peak Fy and the average Fx by the unforced case values, and plot vs. phase angle.



As you can see, there is an optimum phase angle around 240 degrees that minimizes the drag, and reduce the normal force oscillations. This minimum may be in a different spot based on your POD modes. Note that this is a very simple controller, and there was no effort made to adjust the controller based on the shifting mean mode.